(2)

PATENT ABSTRACTS OF JAPAN

(11)Publication number : 11-317014
(43)Date of publication of application : 16.11.1999

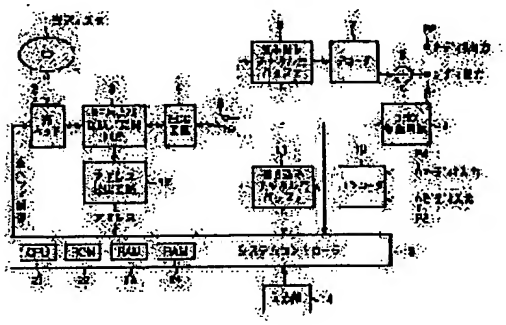
(51)Int.Cl. G11B 20/10
G11B 19/02

(21)Application number : 10-120393
(22)Date of filing : 30.04.1998

(71)Applicant : SONY CORP
(72)Inventor : FUJINAMI YASUSHI
HAMADA TOSHIYA

(54) RECORDING AND REPRODUCING DEVICE AND METHOD, AND PROVIDING MEDIUM

(57)Abstract:
PROBLEM TO BE SOLVED: To enable the securing of compatibility by recording data to a recording medium, judging whether the data are continuously reproducable or not with respect to already recorded data and executing processings concerning the continuous reproduction of the data corresponding to the result.
SOLUTION: After the video signal inputted from an input terminal P3 and the audio signal inputted from an input terminal P4 are encoded in an encoder 10, the encoded data are transmitted to a buffer for write channel 11. The data read out from the buffer 11 are inputted to an ECC circuit 4 via a switch 5 and after the data are added with error correcting codes in the circuit, the signal outputted from an RF and modulation/demodulation circuit 3 is written on an optical disk 1 with an optical head 2. An address detecting circuit 12 detects address information of a track performing a recording to the disk 1 or performing a reproduction from the disk 1 and a CPU 21 finely adjusts the position of the optical head 2 based on the detection result of the address detecting circuit 12.



LEGAL STATUS

[Date of request for examination]

[Date of sending the examiner's decision of rejection]

[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]

[Date of final disposal for application]

[Patent number]

[Date of registration]

[Number of appeal against examiner's decision of rejection]

[Date of requesting appeal against examiner's decision of rejection]

[Date of extinction of right]

JP11-317014

* NOTICES *
JPO and NCIPI are not responsible for any
damages caused by the use of this translation.
1.This document has been translated by computer. So the translation may not reflect the original precisely.
2.**** shows the word which can not be translated.
3.In the drawings, any words are not translated.

CLAIMS

[Claim(s)]
[Claim 1] In the record regenerative apparatus which records or reproduces data to a record medium A record means to record said data to said record medium, and when said data are recorded on said record medium, The record regenerative apparatus characterized by having a judgment means to judge whether it is continuously refreshable to the data already recorded in the data, and an activation means to perform processing about continuation playback of data corresponding to the judgment result of said judgment means.
[Claim 2] Said activation means is a record regenerative apparatus according to claim 1 characterized by changing the record location of said data when said judgment means judges with said continuation playback being impossible.
[Claim 3] Said activation means is a record regenerative apparatus according to claim 1 characterized by recording the continuation playback information that it expresses that continuation playback is impossible on said record medium when said judgment means judges with said continuation playback being impossible.
[Claim 4] In the record playback approach of the record regenerative apparatus which records or reproduces data to a record medium The record step which records said data to said record medium, and when said data are recorded on said record medium, The record playback approach characterized by including the judgment step which judges whether it is continuously refreshable to the data already recorded in the data, and the execute step which performs processing about continuation playback of data corresponding to the judgment result in said judgment step.
[Claim 5] The record step which records said data on the record regenerative apparatus which records or reproduces data to a record medium to said record medium, and when said data are recorded on said record medium, The judgment step which judges whether it is continuously refreshable to the data already recorded in the data, The offer medium characterized by offering the program which performs processing containing the execute step which performs processing about continuation playback of data corresponding to the judgment result in said judgment step.
[Claim 6] In the record regenerative apparatus which records or reproduces data to a record medium An extract means to extract the continuation playback information that it expresses whether continuation playback of data to said data reproduced by playback means to reproduce the data currently recorded on said record medium, and said playback means is possible, The record regenerative apparatus characterized by having an addition means to add a gap to the data reproduced by said playback means, corresponding to said continuation playback information extracted by said extract means.
[Claim 7] In the record playback approach of the record regenerative apparatus which records or

reproduces data to a record medium The extract step which extracts the continuation playback information that it expresses whether continuation playback of data to said data reproduced at the playback step which reproduces the data currently recorded on said record medium, and said playback step is possible, The record playback approach characterized by including the addition step which adds a gap to the data reproduced at said playback step corresponding to said continuation playback information extracted at said extract step.
[Claim 8] The playback step which reproduces the data currently recorded on the record regenerative apparatus which records or reproduces data to a record medium by said record medium, The extract step which extracts the continuation playback information that it expresses whether continuation playback of data to said data reproduced at said playback step is possible, The offer medium characterized by offering the program which performs processing containing the addition step which adds a gap to the data reproduced at said playback step corresponding to said continuation playback information extracted at said extract step.

DETAILED DESCRIPTION

[Detailed Description of the Invention]
[0001]
[Field of the Invention] This invention relates to an offer medium at the record regenerative apparatus which controlled what a user takes for failure in relation to continuation playback of data especially about an offer medium in a record regenerative apparatus and an approach, and a list and an approach, and a list.
[0002]
[Description of the Prior Art] A video data, audio data, etc. set time amount on a disk intermittently, and they are not only recorded continuously, but may be recorded on it. Moreover, the once recorded data may be eliminated and other data may be overwritten.
[0003]
[Problem(s) to be Solved by the Invention] If such elimination or overwrite processing is repeated and performed, the data which should be reproduced continuously may not necessarily be recorded on the location where it continues on a disk, but may be recorded on the location estranged on the disk. In such a case, if the capacity of the buffer at the time of playback of equipment is not enough, depending on the record location, data cannot be reproduced continuously, but there is a case so that playback data may be temporarily missing.
[0004] However, since it was influenced by the capacity of the buffer of equipment, whether a discontinuous part occurs could not secure compatibility between equipment, but when the worst, it had a possibility that a user might take for equipment having broken down.
[0005] This invention is made in view of such a situation, the compatibility of equipment is secured, and it controls that a user takes for equipment having broken down.
[0006]
[Means for Solving the Problem] A record regenerative apparatus according to claim 1 is characterized by to have a record means record data to a record medium, a judgment means judge whether it is continuously refreshable to the data already recorded in the data when data are recorded on a record medium, and an activation means perform processing about continuation playback of data corresponding to the judgment result of a judgment means.
[0007] The record playback approach according to claim 4 is characterized by to be included the record step which records data to a record medium, the judgment step which judges whether it is

continuously refreshable to the data already recorded in the data when data are recorded on a record medium, and the execute step which performs processing about continuation playback of data corresponding to the judgment result in a judgment step.
[0008] When an offer medium according to claim 5 records the record step which records data on the record regenerative apparatus which records or reproduces data to a record medium to a record medium, and data on a record medium, The judgment step which judges whether it is continuously refreshable to the data already recorded in the data, It is characterized by offering the program which performs processing containing the execute step which performs processing about continuation playback of data corresponding to the judgment result in a judgment step.
[0009] It carries out having an extract means extract the continuation playback information express that continuation playback of data to the data reproduced by playback means reproduce the data currently recorded on a record medium, and the playback means is possible for a record regenerative apparatus according to claim 6, and an addition means add a gap to the data reproduced by the playback means corresponding to the continuation playback information extracted by the extract means as the description.
[0010] It carries out containing the extract step which extracts the continuation playback information express that continuation playback of data to the data reproduced at the playback step which reproduces the data currently recorded on a record medium, and a playback step is possible for the record playback approach according to claim 7, and the addition step which add in a gap to the data reproduced at a playback step corresponding to the continuation playback information extracted at an extract step as the description.
[0011] The playback step to which an offer medium according to claim 8 reproduces the data currently recorded on the record regenerative apparatus recorded or reproduced by the record medium in data to a record medium, The extract step which extracts the continuation playback information that it expresses whether continuation playback of data to the data reproduced at the playback step is possible, It is characterized by offering the program which performs processing containing the addition step which adds a gap to the data reproduced at the playback step corresponding to the continuation playback information extracted at the extract step.
[0012] In a record regenerative apparatus according to claim 1, the record playback approach according to claim 4, and an offer medium according to claim 5, processing about continuation playback of data is performed to the data already recorded corresponding to a continuously refreshable judgment result.
[0013] In a record regenerative apparatus according to claim 6, the record playback approach according to claim 7, and an offer medium according to claim 8, a gap is added to data corresponding to the continuation playback information reproduced from the record medium.
[0014]
[Embodiment of the Invention] Although the gestalt of operation of this invention is explained below, it is as follows, when the gestalt (however, an example) of operation [/ in the parenthesis after each means] is added and the description of this invention is described, in order to clarify response relation between each means of invention given in a claim, and the gestalt of the following operations. However, of course, this publication does not mean limiting to what indicated each means.
[0015] When a record regenerative apparatus according to claim 1 records a record means (for example, optical head 2 of drawing 25) to record data to a record medium, and data on a record medium, A judgment means to judge whether it is continuously refreshable to the data already recorded in the data (for example, step S2 of drawing 48), It is characterized by having an

activation means (for example, step S5 of drawing 48) to perform processing about continuation playback of data, corresponding to the judgment result of a judgment means.
[0016] A playback means to reproduce the data with which the record regenerative apparatus according to claim 6 is recorded on the record medium (for example, optical head 2 of drawing 25), An extract means to extract the continuation playback information that it expresses whether continuation playback of data to the data reproduced by the playback means is possible (for example, step S41 of drawing 50), It is characterized by having an addition means (for example, step S46 of drawing 50) to add a gap to the data reproduced by the playback means, corresponding to the continuation playback information extracted by the extract means.
[0017] The file arrangement on the record medium (media) with which it sets to this invention first, and information is recorded or reproduced is explained. On media, as shown in drawing 1 , seven kinds of files as follows are recorded.
VOLUME.TOCALBUM.STRPROGRAM_$$$.PGITITLE_###.VDRCHUNKGROUP_@@@.CGITCHUNK_%%%%.ABSTCHUNK_%%%%.MPEG 2[0018] VOLUME.TOC and ALBUM.STR are put on a root directory. Moreover, "PROGRAM_$$$.PGI" ("$$$" expresses a program number here) is put on directory "PROGRAM" directly under a root directory. Similarly, "CHUNKGROUP_@@@.CGIT" ("@@@" expresses the chunk group number here) is put on directory "CHUNKGROUP", and "CHUNK_%%%%.ABST" ("%%%%" expresses a chunk number here) is put on directory "CHUNK" for "TITLE_###.VDR" ("###" expresses a title number here) by directory "TITLE" directly under a root directory, respectively.
[0019] One more or more subdirectories are created by the MPEGAV directory directly under a root directory, and "CHUNK_%%%%.MPEG 2" (%%%% expresses a chunk number here) is put on the bottom of it.
[0020] As for the file of VOLUME.TOC, it is common that it is on [one] media. However, by the media of special structures, such as media of the hybrid construction of ROM and RAM, it is also possible that more than one exist. This file is used in order to show the property of the whole media.
[0021] The structure of VOLUME.TOC is shown in drawing 2 . file_type_id is placed at a head and it is shown that an applicable file is VOLUME.TOC by this. Next, volume_information() continues and, finally text_block() continues.
[0022] The configuration of volume_information() is shown in drawing 3 . This contains volume_attribute(), resume(), volume_rating(), write_protect(), play_protect(), and recording_timer().
[0023] volume_attribute() is a field which records the attribute of logical volume, and the detailed structure is shown in drawing 4 . As shown in this drawing, title_playback_mode_flag, program_playback_mode_flag, etc. are contained in this field.
[0024] resume() is a field which records the information for restoring the condition in front of eject at the time of the reinsertion of media, and the detailed structure is shown in drawing 5 .
[0025] volume_rating() of drawing 3 is a field which records the information for realizing Parental Control/Rating to the whole volume according to age or a category, and the detailed structure is shown in drawing 6 .
[0026] write_protect() of drawing 3 is a field which records modification to title currently recorded in volume, and program, and the information which restricts elimination actuation, and the detailed structure is shown in drawing 7 .
[0027] play_protect() of drawing 3 is a field which records setting out of the playback authorization to title currently recorded in volume, and program, and disapproval, or the

information which restricts the count of playback, and the detailed structure is shown in drawing 8 .

[0028] recording_timer() of drawing 3 is a field which records the information which controls chart lasting time, and the detailed structure is shown in drawing 9 .

[0029] The detailed structure of text_block() of VOLUME.TOC of drawing 2 is shown in drawing 10 . language_set() and text_item are contained in this text_block() and that detailed structure is shown in drawing 11 and drawing 12 , respectively.

[0030] As for the file of ALBUM.STR of drawing 1 , it is common that it is on [one] media. However, by the media of special structures, such as media of the hybrid construction of ROM and RAM, it is also possible that more than one exist. Combining two or more media, this file is used in order to make it a configuration which is one media.

[0031] The structure of this ALBUM.STR is shown in drawing 13 . file_type_id is placed at a head and it is shown that an applicable file is ALBUM.STR. Next, album() continues and, finally text_block() continues.

[0032] album() is a field which records the information for treating two or more volume(s) (two or more media) as one settlement, and the detailed structure is shown in drawing 14 .

[0033] As for the file of TITLE_###.VDR of drawing 1 , only the number of titles exists. A title means one music said by compact disc, and one program of television broadcasting. The structure of this information is shown in drawing 15 . file_type_id is placed at a head and it is shown that an applicable file is TITLE_###.VDR by this. Next, title_info() continues and, finally text_block() continues. # ## is a character string which shows a title number.

[0034] title_info() is a field for recording the start point of title on chunkgroup, an ending point, and the other attributes about title, and the detailed structure is shown in drawing 16 .

[0035] As for the file of PROGRAM_$$$.PGI of drawing 1 , only the number of programs exists. A program consists of two or more cuts which specified some (or all) fields of a title, and each cut is reproduced in the specified sequence. The structure of this information is shown in drawing 17 . file_type_id is placed at a head and it is shown that an applicable file is PROGRAM_$$$.PGI. Next, program() continues and, finally text_block() continues. $$$ is a character string which shows a title number.

[0036] program() is a field which records information required [without performing irreversible edit to a raw material] to collect the required parts of title and reproduce, and the detailed structure is shown in drawing 18 .

[0037] program() of drawing 18 has one play_list. The detail of this play_list() is shown in drawing 19 .

[0038] Two or more play_item() is put on play_list. The detail of play_item() is shown in drawing 20 .

[0039] As for the file of CHUNKGROUP_@@@.CGIT of drawing 1 , only the number of chunk groups exists. A chunk group is the DS for putting a bit stream in order. This file is not recognized by the user at the part which is operating ordinarily the equipment with which a user does record playback of the media, such as VDR (video disc recorder).

[0040] The structure of this information is shown in drawing 21. file_type_id is placed at a head and it is shown that an applicable file is CHUNKGROUP_@@@.CGIT. chunkgroup_time_base_flags and chunkgroup_time_base_offset are in the degree, chunk_connection_info() continues a degree and, finally text_block() continues.

[0041] chunkgroup_time_base_flags shows flag about the criteria counter of chunkgroup, and chunkgroup_time_base_offset shows the start time of the conventional-time shaft in chunkgroup.

This is a value set to the counter counted up by 90kHz, and has the magnitude of 32 bits. chunk_connection_info() is a field which memorizes the information on unique points, such as a switching point of video, and a synchronization of video and audio, and the detailed structure is shown in drawing 22 .

[0042] The loop formation of chunk_arrangement_info() is put on this chunk_connection_info() only for the number of the chunks belonging to a chunk group. The detail of this chunk_arrangement_info() is shown in drawing 23 .

[0043] As for the file of CHUNK_%%%%.ABST of drawing 1 , only the number of chunks exists. A chunk is an information file corresponding to one stream file. The structure of this information is shown in drawing 24 . file_type_id is placed at a head and, thereby, it is shown that an applicable file is CHUNK_%%%%.ABST.

[0044] The file of CHUNK_%%%%.MPEG 2 of drawing 1 is a stream file. This file stores the bit stream of MPEG and differs from other files recording only information.

[0045] Drawing 25 expresses the example of a configuration of the optical disk unit which records or reproduces information to the optical disk as media which have the above files. In this optical disk unit, one optical head 2 is formed to the erasable optical disk 1 of one sheet, and this optical head 2 is shared by read-out of data and the both sides of writing.

[0046] after getting over in RF, and the recovery/modulation circuit 3, an error correction is performed in the ECC circuit 4, and reading appearance of the bit stream in which reading appearance was carried out by the optical head 2 from the optical disk 1 is carried out for carrying out reading appearance through a switch 5, and absorbing the difference of a rate and a decoding rate, and it is sent to the buffer 6 for channels. The output of the buffer 6 for read-out channels is supplied to decoding 7. The buffer 6 for read-out channels is constituted so that R/W may be possible from a system controller 13.

[0047] The bit stream outputted from the buffer 6 for read-out channels is decoded by the decoder 7, and a video signal and an audio signal are outputted from there. The video signal outputted from the decoder 7 is inputted into the synthetic circuit 8, and is outputted and displayed on the display which is not illustrated from an output terminal P1 after the video signal which the 0SD (On Screen Display) control circuit 9 outputs, and compounded **. The audio signal outputted from the decoder 7 is sent to the loudspeaker which is not illustrated from an output terminal P2, and is reproduced.

[0048] On the other hand, after the video signal inputted from the input terminal P3 and the audio signal inputted from the input terminal P4 are encoded with an encoder 10, it is sent to the buffer 11 for write-in channels for writing in with an encoding processing rate and absorbing a difference with a rate. It is constituted so that the R/W also of this buffer 11 for write-in channels may be possible from a system controller 13.

[0049] Reading appearance of the data stored in the buffer 11 for write-in channels is carried out from the buffer 11 for write-in channels, and after being inputted into the ECC circuit 4 through a switch 5 and adding an error correcting code, they are modulated in RF, and the recovery/modulation circuit 3. The signal (RF signal) outputted from RF, and the recovery/modulation circuit 3 is written in an optical disk 1 by the optical head 2.

[0050] The address detector 12 detects the address information of the truck which an optical disk 1 records or reproduces. A system controller 13 controls actuation of each part of this optical disk unit, and has RAM23 for storing temporarily ROM22 which stored the processing program which CPU21 and CPU21 which perform various kinds of control should perform, the data produced in the processing process, and RAM24 which memorizes various kinds of information

files recorded or reproduced to an optical disk 1. CPU21 tunes the location of the optical head 2 finely based on the detection result of the address detector 12. CPU21 performs change control of a switch 5 again. The input section 14 which consists of various kinds of switches, a carbon button, etc. is operated by the user when inputting various kinds of commands.

[0051] Next, reading actuation of a fundamental information file is explained. For example, when reading an information file, the file system operating instructions beforehand included in the processing program are used for CPU21 of the stem controller 13, and the physical address and die length on the "VOLUME.TOC" optical disk 1 with which "VOLUME.TOC" is recorded are decided. Then, CPU21 reads the optical head 2 and is made to move it to a location based on the address information of this "VOLUME.TOC." And after CPU21 reads a switch 5, changes it to the buffer 6 side for channels and tunes the location of the optical head 2 finely further while it reads the ECC circuit 4 to the optical head 2, RF and the recovery/modulation circuit 3, and a list and sets it as the mode, it makes read-out by the optical head 2 start. After reading appearance of the content of "VOLUME.TOC" is carried out by the optical head 2 by this, it gets over by RF, and the recovery/modulation circuit 3 and an error correction is further performed by the ECC circuit 4, it is accumulated in the buffer 6 for read-out channels.

[0052] As for CPU21, read-out is stopped, when the amount of data accumulated in the buffer 6 for read-out channels is equal to the magnitude of "VOLUME.TOC" or becomes larger. Then, CPU21 reads applicable data from the buffer 6 for read-out channels, and RAM24 is made to memorize it.

[0053] Next, the case where a "VOLUME.TOC" information file is written in is explained as an example about fundamental information file write-in actuation. The file system operating instructions beforehand included in the processing program are used for CPU21, and it is equal to "VOLUME.TOC" which it is going to write into a file system (optical disk 1) after this, or looks for a free area with larger magnitude, and decides the address.

[0054] Next, CPU21 transmits "VOLUME.TOC" which is prepared for RAM24 and which should newly be written in to the buffer 11 for write-in channels. Then, CPU21 writes in the optical head 2 and is made to move it to a location based on the address information of a free area. And after CPU21 writes in a switch 5, changes it to the buffer 11 side for channels and tunes the location of the optical head 2 finely while it sets the ECC circuit 4 as a write mode at the optical head 2, RF and the recovery/modulation circuit 3, and a list, it makes the writing by the optical head 2 start.

[0055] After reading appearance of the content of "VOLUME.TOC" which this newly prepared is carried out from the buffer 11 for write-in channels, being inputted into the ECC circuit 4 through a switch 5 and adding an error correcting code, RF, and the recovery/modulation circuit 3 become irregular. The signal outputted from RF, and the recovery/modulation circuit 3 is recorded on an optical disk 1 by the optical head 2. Reading appearance is carried out from the buffer 11 for write-in channels, and when the amount of data recorded on the optical disk 1 becomes equal to the magnitude of "VOLUME.TOC", as for CPU21, write-in actuation is stopped.

[0056] Finally, the file system operating instructions beforehand included in the processing program are used for CPU21, and it rewrites them so that it may point to the location which wrote in newly the pointer indicating "VOLUME.TOC" in a file system (optical disk 1).

[0057] Next, the case where a stream called CHUNK_0001.MPEG 2 of drawing 1 is reproduced about fundamental in-stream playback actuation is explained as an example. The file system operating instructions beforehand included in the processing program are used for CPU21, and

the physical address and die length on the optical disk 1 with which "CHUNK_0001.MPEG 2" is recorded are decided. Then, CPU21 reads the optical head 2 and is made to move it to a location based on the address information of this "CHUNK_0001.MPEG 2." And while reading the ECC circuit 4 to the optical head 2, RF and the recovery/modulation circuit 3, and a list and setting it as the mode, after reading a switch 5, changing to the buffer 6 side for channels and tuning the location of the optical head 2 finely, read-out by the optical head 2 is made to start.

[0058] The content of "CHUNK_0001.MPEG 2" in which reading appearance was carried out by the optical head 2 reads to RF and the recovery/modulation circuit 3, the ECC circuit 4, and a list through a switch 5, and is accumulated in the buffer 6 for channels. The data stored in the buffer 6 for read-out channels are outputted to a decoder 7, decoding is performed and a video signal and an audio signal are outputted, respectively. An audio signal is outputted from an output terminal P2, and a video signal is outputted from an output terminal P1 through the synthetic circuit 8.

[0059] It is read from an optical disk 1, and when the amount of data decoded and displayed became equal to the magnitude of "CHUNK_0001.MPEG 2", or when a halt of read-out actuation is specified from the input section 14, as for CPU21, read-out and decoding are stopped.

[0060] Next, the case where a "CHUNK_0001.MPEG 2" information file is written in for fundamental stream record actuation is explained as an example. The file system operating instructions beforehand included in the processing program are used for CPU21, and it is equal to "CHUNK_0001.MPEG 2" which it is going to write into a file system (optical disk 1) after this, or looks for a free area with larger magnitude than it, and decides the address.

[0061] After the video signal inputted from the input terminal P3 and the audio signal inputted from the input terminal P4 are encoded by the encoder 10, it is accumulated in the buffer 11 for write-in channels. Then, CPU21 writes in the optical head 2 and is made to move it to a location based on the address information of a free area. And after CPU21 writes in a switch 5, changes it to the buffer 11 side for channels and tunes the location of the optical head 2 finely while it sets the ECC circuit 4 as a write mode at the optical head 2, RF and the recovery/modulation circuit 3, and a list, it makes the writing by the optical head 2 start. Reading appearance of the content of "CHUNK_0001.MPEG 2" which this newly prepared is carried out from the buffer 11 for write-in channels, it is inputted into the optical head 2 through a switch 5, the ECC circuit 4, RF, and the recovery/modulation circuit 3, and is recorded on an optical disk 1.

[0062] Reading appearance is carried out from the buffer 11 for write-in channels, and when the amount of data recorded on the optical disk 1 becomes equal to the value set up beforehand, or when a halt of write-in actuation is specified from the input section 14, as for CPU21, write-in actuation is stopped. Finally, the file system operating instructions beforehand included in the processing program are used for CPU21, and it rewrites them so that it may point to the location which wrote in newly the pointer indicating "CHUNK_0001.MPEG 2" in a file system (optical disk 1).

[0063] Now, an information file and a stream file as shown in an optical disk 1 at drawing 26 shall be recorded. In this example, the file of one program of the identifier of "PROGRAM_001.PGI" is included. Moreover, the file of three titles of the identifier of "TITLE_001.VDR", "TITLE_002.VDR", and "TITLE_003.VDR" is included in this optical disk 1.

[0064] Furthermore, the file of the two chunk groups "CHUNKGROUP_001.CGIT" and "CHUNKGROUP_002.CGIT" is included in this optical disk 1. Moreover, while the file of three

streams of the identifier of "CHUNK_0001.MPEG 2", "CHUNK_0011.MPEG 2", and "CHUNK_0012.MPEG 2" is included, three information files, "CHUNK_0001.ABST", "CHUNK_0011.ABST", and "CHUNK_0012.ABST", are put on this optical disk 1 as information corresponding to each.

[0065] The logical structure of the optical disk 1 which has the information file shown in drawing 26 and a stream file comes to be shown in drawing 27. this example -- a chunk information file -- "CHUNK_0001.ABST "stream file" CHUNK_0001.MPEG 2" -- moreover, a chunk information file -- "CHUNK_0011.ABST "stream file" CHUNK_0011.MPEG 2" -- further -- a chunk information file -- "CHUNK_0012.ABST "stream file" CHUNK_0012.MPEG 2" is specified, respectively. It is the field called chunk_file_id in CHUNK_%%%%.ABST of drawing 24 , and, specifically, the file ID of a stream is specified.

[0066] furthermore -- this example -- a chunk group information file -- "CHUNKGROUP_001.CGIT "chunk information file" CHUNK_0001.ABST" -- moreover, a chunk group information file -- "CHUNKGROUP_002.CGIT "chunk information file" CHUNK_0011.ABST" and "CHUNK_0012.ABST" are specified, respectively. Specifically, the file ID of chunk information is specified in the field called chunk_info_file_id in chunk_arrangement_info() of drawing 23 . This chunk_arrangement_info() is in a chunk group information file, and has DS in which only the number of the chunks belonging to an applicable chunk group exists (chunk_arrangement_info() of drawing 23 is described by chunk_connection_info() of drawing 22 , and this chunk_connection_info() is described by CHUNKGROUP_###.CGIT of drawing 21).

[0067] There is only one chunk_arrangement_info() in CHUNKGROUP_001 and chunk_info_file_id in it specifies CHUNK_0001 as them. CHUNK_0011 and CHUNK_0012 are specified for chunk_arrangement_info() as CHUNKGROUP_002 those with two, and in it, respectively. For the reason in such a case, a chunk group can specify now the playback sequence of two or more chunks etc.

[0068] Specifically, the initial value of the clock in an applicable chunk group is first defined by chunkgroup_time_base_offset in CHUNKGROUP_###.CGIT of drawing 21 . Next, in case each chunk is registered, presentation_start_cg_count and presentation_end_cg_time_count of chunk_arrangement_info() of drawing 23 are specified.

[0069] For example, as shown in drawing 28 , A and the die length (time amount) of CHUNK_0012 are set to B for the die length (time amount) of CHUNK_0011. presentation_start_cg_count of CHUNK_0011 is equal to chunkgroup_time_base_offset and presentation_end_cg_count is equal to "chunk_group_time_base_offset+A". Moreover, presentation_start_cg_count of CHUNK_0012 is equal to chunkgroup_time_base_offset+A and presentation_end_cg_count is equal to "chunk_group_time_base_offset+A+B". Thus, setting out defines CHUNKGROUP_002 as what reproduced CHUNK_0011 and CHUNK_0012 continuously.

[0070] In addition, when a lap is in the playback time of day of CHUNK_0011 and CHUNK_0012, assignment is possible by shifting time of day such. Moreover, in transition between two streams, special effect (fade-in, fade-out, wipe, etc.) can be specified now by describing to transition_info() in chunk_arrangement_info() of drawing 23 .

[0071] In the example of drawing 26 (drawing 27), "TITLE_002.VDR "chunk group information file" CHUNKGROUP_001.CGIT "title information file [moreover,]" TITLE_003.VDR "chunk group information file" CHUNKGROUP_002.CGIT" is specified as title information file"TITLE_001.VDR", respectively. It is the field called cgit_file_id and a

chunk group's file ID was specified, it is the field called title_start_chunk_group_time_stamp and title_end_chunk_group_time_stamp further, and, specifically, the time range where an applicable title is defined within a chunk group is specified [be / it / under / title_info() / of drawing 16 / setting].

[0072] For example, in the example of drawing 27 , TITLE_001 point to the first half of CHUNKGROUP_001, and TITLE_002 are pointing to the second half, respectively. In addition, this division is performed by the demand from a user, and for a user, that location is arbitrary and cannot be decided beforehand. Suppose that it was set as the location where only A separated the location of division by TITLE_001 and TITLE_002 from the head of CHUNKGROUP_001 here.

[0073] TITLE_001 specify CHUNKGROUP_001 as a chunk group, as start time of a title, specify the start time of CHUNKGROUP_001 and specify the time of day of a point specified by the user as end time of a title.

[0074] That is, as title_start_chunk_group_time_stamp of TITLE_001, chunkgroup_time_base_offset (top location) of CHUNKGROUP_001 is set up and what applied the die length of A to chunkgroup_time_base_offset of CHUNKGROUP_001 is set up as title_end_chunk_group_time_stamp of TITLE_001.

[0075] Moreover, TITLE_002 specify CHUNKGROUP_001 as a chunk group, as start time of a title, specify the time of day of a point specified by the user, and specify the end time of CHUNKGROUP_001 as end time of a title.

[0076] That is, as title_start_chunk_group_time_stamp of TITLE_002, what applied the die length of A to chunkgroup_time_base_offset (top location) of CHUNKGROUP_001 is set up, and what applied the die length of CHUNKGROUP_001 to chunkgroup_time_base_offset of CHUNKGROUP_001 is set up as title_end_chunk_group_time_stamp of TITLE_002.

[0077] Furthermore, TITLE_003 specify CHUNKGROUP_002 as a chunk group, specify the start time of CHUNKGROUP_002 as start time of a title, and specify the end time of CHUNKGROUP_002 as end time of a title.

[0078] That is, as title_start_chunk_group_time_stamp of TITLE_003, chunkgroup_time_base_offset of CHUNKGROUP_002 is set up and what applied the die length of CHUNKGROUP_002 to chunkgroup_time_base_offset of CHUNKGROUP_002 is set up as title_end_chunk_group_time_stamp of TITLE_003.

[0079] Furthermore, in this example, program information file"PROGRAM_001.PGI" specifies a part of TITLE_001 and a part of TITLE_003 that it reproduces in this sequence. A title is specified by title_number in play_item() of drawing 20 , it is that the time of day defined by each title defines a start point and an ending point, and, specifically, one cut is extracted. Two or more such cuts are collected and a program is constituted.

[0080] Next, the actuation in the case of carrying out ****** (appending record) of the new information to an optical disk 1 is explained. Specifically, this record is performed timed recording or by a user's operating the input section 14 and ordering real time an image transcription to an optical disk unit. When an image transcription carbon button is pushed in the case of the latter, image transcription end time cannot be predicted, but end time can be predicted when the carbon button of an one-touch image transcription function (function in which an image transcription is performed only for fixed time amount after actuation) is pushed.

[0081] Here, it explains taking the case of timed recording. In this case, the user of an optical disk unit shall specify in advance image transcription start time, image transcription end time, the bit rate of a bit stream, the channel that performs an image transcription. Moreover, when an

image transcription is reserved, it shall be checked beforehand that the availability corresponding to a bit rate and image transcription time amount is left behind to the optical disk 1.
[0082] When further record is performed to an optical disk 1 between the times of activation of the record reserved as the time of record reservation, there is a case where it becomes impossible to secure the capacity of the part which records the program reserved this time with the specified bit rate. In such a case, CPU21 lowers a bit rate from the specified value, records the information for the reserved time amount, or leaves a bit rate as it is, and records only recordable time amount. When nonconformity appears in the record which further record was performed and reserved CPU21 at this time, it cannot be overemphasized that the message which tells a user that is emitted.
[0083] Now, if the start time of the reserved image transcription draws near, CPU21 will use the timer and clock to build in and will return the mode to a mode of operation automatically from a sleep mode. And CPU21 secures only the field where the file system operating instructions beforehand included in the processing program are used, and the reserved program can record them on an optical disk 1. That is, it is the area size which needs the numeric value which multiplied the result (image transcription time amount) of having subtracted start time from the end time of timed recording by the bit rate to record the reserved program, and, as for CPU21, the field of this magnitude is secured first. In addition, in order to register as a new title, when a title information file etc. is required, it is necessary to secure only the capacity which can record those information files to an optical disk 1, when an information file needs to be recorded on the occasion of this record in addition to a stream file. When the field of a required part is not securable, a response will be taken by approach (approaches, such as modification of a bit rate, and an image transcription only within the time amount which can be recorded on videotape) which was mentioned above.
[0084] In addition, since it is record of a new title at this time, a user attaches the file name of a stream file new as a new stream file of a new stream directory. Here, this is set to \MPEGAV\STREMS_003\CHUNK_0031. That is, as shown in drawing 29 , it considers as the file of the identifier of CHUNK_0031.MPEG 2 under STREAM_003 directory under the MPEGAV directory under a root directory.
[0085] CPU21 orders activation of a recording mode to each part. For example, after the video signal inputted into the input terminal P3 from the tuner which is not illustrated and the audio signal inputted into the input terminal P4 are encoded by the encoder 10, it is accumulated in the buffer 11 for write-in channels. Then, CPU21 writes in the optical head 2 and is made to move it to a location based on the address information of the field secured previously. And after CPU21 writes in a switch 5, changes it to the buffer 11 side for channels and tunes the location of the optical head 2 finely while it sets the ECC circuit 4 as a write mode at the optical head 2, RF and the recovery/modulation circuit 3, and a list, it makes the writing by the optical head 2 start. Reading appearance of the content of "CHUNK_0031.MPEG 2" which this newly prepared is carried out from the buffer 11 for write-in channels, and it is recorded on a switch 5, the ECC circuit 4, RF and the recovery/modulation circuit 3, and a list by the optical disk 1 through the optical head 2.
[0086] When the above write-in actuation is continued and one of the following conditions occurs, CPU21 stops write-in actuation.
1) It is [0087], when the end time of the reserved record comes, record becomes impossible to an optical disk 1 according to the cause of the lack of 2 capacity, and others and it is ordered in a halt of 3 image-transcription actuation. Next, the file system operating instructions beforehand

included in the processing program are used for CPU21, and it rewrites them to the value indicating the location which wrote in newly the pointer indicating "CHUNK_0031.MPEG 2" in a file system. Moreover, CPU21 prepares each file of chunk information, chunk group information, and title information, and names and records an appropriate identifier. In addition, it is necessary to secure only the availability which can record these files on an optical disk 1 at the time of record or reservation.

[0088] Thus, as shown, for example in drawing 30 , a new information file is created. In this drawing, what attached the asterisk (*) to the right shoulder of a file name is the file newly created this time.

[0089] Drawing 31 shows the relation of the newly done information file. TITLE_004 specified CHUNKGROUP_003, CHUNKGROUP_003 specified CHUNK_0031 and CHUNK_0031 specify STREAM_0031.

[0090] That is, a new stream is registered into the information file as TITLE_004. By the function to check the title of an optical disk unit, a user can know the attribute of TITLE_004 etc. and can reproduce TITLE_004.

[0091] Next, the actuation in the case of carrying out overwrite record is explained on the optical disk 1 which is illustrated to drawing 26 (drawing 27). The thing of the actuation which records a new (eliminating the program) program on the program currently recorded by then like the case where a signal is recorded on a video tape is called overwrite record.

[0092] The location which starts overwrite record in overwrite record is important. For example, starting overwrite record from the head of TITLE_001 presupposes from a user that it was specified. Overwrite record is performed at this time, rewriting TITLE_001, TITLE_002, and TITLE_003 in order, respectively. Even if it rewrites to the last of TITLE_003, when record actuation is not completed yet, a new field is secured to the free area on an optical disk 1, and record is continued. For example, since TITLE_001 are located before a recording start location when TITLE_002 are made into a recording start location, it is not rewritten by this record actuation.

[0093] Now, it shall overwrite by timed recording from the head of TITLE_003. In this case, the user of an optical disk unit shall specify in advance image transcription start time, end time, the bit rate of a bit stream, the channel that performs an image transcription. Moreover, in overwrite record, the important recording start location should be specified as the head of TITLE_003. When an image transcription is furthermore reserved in this case, it shall be checked beforehand that the capacity corresponding to a bit rate and image transcription time amount exists on an optical disk 1. The sum of the total capacity of the title which can overwrite from the specified location in overwrite record (plurality), and the availability of an optical disk 1 serves as record possible capacity. That is, in this case, it becomes stream STREAM_0011 which TITLE_003 manage, and the capacity which can record the sum of the availability on STREAM_the total capacity of 0012 and an optical disk 1.

[0094] In overwrite record, there is some alternative in what kind of sequence to perform actual record, to a recordable capacitive component. The approach of recording in order of the stream specified in the title is thought first. That is, when continuing record from the head of STREAM_0012 if record is first started from the head of STREAM_0011 in this case and it records on it to the end of STREAM_0011, and recording to the end of STREAM_0012, it is the approach of recording on a free area shortly. Another approach is the approach of recording on the existing stream, when it records on a free area and a free area is lost first.

[0095] The former approach is excellent in the semantics of the emulation of a video tape. That

is, it has the description that a user is easy to be understood, for the purpose of being the same actuation as a video tape. Since elimination of the stream already recorded is carried out to deferment, although the latter approach is recorded, it has the description referred to as excelling in respect of protection.

[0096] In addition, when further record is performed to an optical disk 1 between the times of activation of the record reserved as the time of record reservation, capacity of the part which records the program reserved this time with the specified bit rate may be unable to be secured. In such a case, like the case where it mentions above, a bit rate is automatically lowered at the time of reservation activation, all are recorded by the reserved time amount, or a bit rate is left as it is and record is performed only for recordable time amount.

[0097] If the start time of the reserved image transcription draws near, an optical disk unit will return to a mode of operation from a sleep mode. CPU21 secures all the availabilities on an optical disk 1. Of course, although there is also a method of securing when an availability was not secured but it is needed at this event, a field required before a recording start shall be secured here for explanation.

[0098] in addition, when a required area size is known beforehand, you may make it only the part which added that margin a little or -- secure capacity on timed recording etc., only as for a required part, since start time, end time, and a bit rate are specified When an information file needs to be recorded on the occasion of this record, for example, in order to register as a new title, when a title information file etc. is required, it is necessary to leave only the capacity which can also record those information files.

[0099] Here, the file name of a stream file new as a new stream file of a new stream directory shall be given. That is, a file name is set to \MPEGAV\STREMS_002\CHUNK_0031 here. That is, as shown in drawing 32, the file of the identifier of CHUNK_0031.MPEG 2 under STREAM_002 directory under the MPEGAV directory under a root directory is created.

[0100] After the video signal inputted into the input terminal P3 and the audio signal inputted into the input terminal P4 are encoded by the encoder 10, it is accumulated in the buffer 11 for write-in channels. Then, CPU21 writes in the optical head 2 and is made to move it to a location based on the address information of the field secured previously. And after CPU21 writes in a switch 5, changes it to the buffer 11 side for channels and tunes the location of the optical head 2 finely while it sets the ECC circuit 4 as a write mode at the optical head 2, RF and the recovery/modulation circuit 3, and a list, it makes the writing by the optical head 2 start. Reading appearance of the content of "CHUNK_0031.MPEG 2" which this newly prepared is carried out from the buffer 11 for write-in channels, and it is recorded on a switch 5, the ECC circuit 4, RF and the recovery/modulation circuit 3, and a list by the optical disk 1 through the optical head 2.

[0101] First of all at this time, stream file"CHUNK_0011.MPEG 2" is rewritten. And if record is performed to the last of "CHUNK_0011.MPEG 2" next, record will be advanced to "CHUNK_0012.MPEG 2" and record will be further advanced to "CHUNK_0031.MPEG 2."

[0102] Like the case where the above actuation is continued and mentioned above, when either of three conditions occurs, as for CPU21, write-in actuation is stopped.

[0103] Next, the file system operating instructions beforehand included in the processing program are used for CPU21, and it updates a stream file, chunk information, chunk group information, and title information.

[0104] By the way, the configuration of a file changes with the timing which writing ended. For example, when record is further performed to CHUNK_0031.MPEG 2 after ending overwrite of two streams, CHUNK_0011.MPEG 2 and CHUNK_0012.MPEG 2, the configuration of the file

of an optical disk 1 comes to be shown in drawing 33 . What attached the asterisk (*) to the right shoulder of a file name is the file newly created this time.
[0105] Drawing 34 shows the relation of the file (file of drawing 33) which did in this way and was newly done. As compared with drawing 31 , CHUNK_0031 are increasing as CHUNK contained in CHUNKGROUP_002 specified by TITLE_003, and CHUNK_0031 specify STREAM_0031 so that clearly.
[0106] On the other hand, when overwrite record is completed in the middle of overwrite of the existing stream (for example, when overwrite record is completed while having been record of CHUNK_0011), since it was not overwritten, the stream of CHUNK_0031 secured for overwrite is opened. In this case, processing of a special title is performed. That is, when overwrite record is started from the head of TITLE_003 and record is completed by the middle, a title is divided there. That is, as shown in drawing 35 , it is referred to as TITLE_003 with from an overwrite recording start location to a new termination location, and the (remaining part of TITLE_003 from the first) after it is set to TITLE_004.
[0107] Next, actuation of title playback is explained. The optical disk 1 which has a file as shown in drawing 26 now shall be inserted in an optical disk unit, and title playback shall be carried out. First, when an optical disk 1 is inserted, CPU21 reads an information file from an optical disk 1, and RAM24 is made to memorize it. This actuation is performed by repeating the reading actuation of a fundamental information file mentioned above.
[0108] CPU21 reads VOLUME.TOC and ALBUM.STR first. Next, it investigates how many CPU21 has the file which has the extension of ".VDR" below in directory "TITLE". The file with this extension is a file with the information on a title, and serves as the number of the number of those files, i.e., a title. The number of titles is set to 3 in the example of drawing 26 . Next, CPU21 reads three title information files, and RAM24 is made to memorize it.
[0109] CPU21 controls the OSD control circuit 9, generates the text which shows the information on the title currently recorded on the optical disk 1, is made to compound with a video signal by the synthetic circuit 8, from an output terminal P1, is made to output to a display and is displayed. In now, there being three titles, and each of the die length and the attributes of three titles (an identifier, recorded time) are displayed.
[0110] Here, a user presupposes that playback of TITLE_002 was specified. The file ID which specifies CHUNKGROUP_001 is recorded on the information file of TITLE_002 (cgit_file_id in title_info() of drawing 16), and CPU21 makes CHUNKGROUP_001 store in RAM24 while memorizing this.
[0111] Next, as for CPU21, it investigates to which CHUNK the start time and end time (title_start_chunk_group_time_stamp and title_end_chunk_group_time_stamp in title_info() of drawing 16) of TITLE_002 correspond. This is performed by comparing the information (presentation_start_cg_time_count and presentation_end_cg_time_count in chunk_arrangement_info() of drawing 23) into which each CHUNK is registered out of the information on CHUNKGROUP. In now, as shown in drawing 27 , as for the start time of TITLE_002, it turns out that close is in the middle of CHUNK_0001. That is, in order to reproduce TITLE_002 from a head, it turns out that it is said that what is necessary is just to start playback from the middle of stream file"CHUNK_0001.MPEG 2."
[0112] Next, CPU21 investigates where [in a stream] the head of TITLE_002 hits. That is, it is calculated how many the start time of TITLE_002 hits as offset time of day in a stream (time stump), then it uses the focus information in a CHUNK file, and the playback start point which hits just before start time is specified. By this, the offset distance from the file head of a playback

start point is able to be decided.
[0113] Next, the file system operating instructions beforehand included in the processing program are used for CPU21, and the physical address and die length on the optical disk 1 with which "CHUNK_0001.MPEG 2" is recorded are decided. Furthermore, the offset address of the playback start point for which it asked previously is added to this address, and the address of the playback start point of TITLE_002 is decided eventually.
[0114] Then, CPU21 reads the optical head 2 and is made to move it to a location based on the address information of this "CHUNK_0001.MPEG 2." And after CPU21 reads a switch 5, changes it to the buffer 6 side for channels and tunes the location of the optical head 2 finely while it reads the ECC circuit 4 to the optical head 2, RF and the recovery/modulation circuit 3, and a list and sets it as the mode, it makes read-out by the optical head 2 start. The content of "CHUNK_0001.MPEG 2" reads by this, and it is accumulated in the buffer 6 for channels.
[0115] The data stored in the buffer 6 for read-out channels are outputted to a decoder 7, decoding is performed and a video signal and an audio signal are outputted. When the amount of data which was read from the optical disk 1, was decoded and was displayed becomes equal to the magnitude of "CHUNK_0001.MPEG 2", CPU21 shifts to playback of TITLE_003. This playback actuation of TITLE_003 is the same actuation as playback actuation of TITLE_002.
[0116] When playback of the title registered is completed, or when a halt of read-out actuation is directed, read-out and decoding are suspended.
[0117] In addition, although CPU21 tends to read VOLUME.TOC and ALBUM.STR when a disk is inserted when a new disk is inserted in an optical disk unit as an optical disk 1, or when the disk in a different format is inserted, such a file will not exist in these disks. In such a case, i.e., when VOLUME.TOC and ALBUM.STR cannot be read, CPU21 outputs a message and asks a user for directions. A user directs to CPU21, he makes it initialize whether an optical disk 1 is made to eject (for example, when it is a disk in a different format), or he restores data by a certain approach (for example, although it is a disk in the same format, when data are destroyed). (for example, when it is a new disk in the same format)
[0118] Next, title is explained further. As shown in drawing 15 , TITLE_###.VDR is file for storing the information on title. The information about one title is recorded on one title_info(). The number of title_info() which exists in TITLE_###.VDR is one. Therefore, only in the number of title(s), TITLE_###.VDR exists in volume.
[0119] title number is not defined in title_info() of drawing 16, but is determined by the file name or file id. Therefore, positive integer ### of the TITLE_###.VDR(s) expresses title number. title is the part of the range from the title index showing a start point attached to chunkgroup to the title index showing the head of the following title, or the range to the ending point of chunkgroup rather than structure.
[0120] As shown in drawing 36 , file_type_id of TITLE_###.VDR of drawing 15 is id which shows that it is file on which title_info() was recorded, and is expressed with the character string of die length 16. text_block() is a field for storing various text(s), and only text item allowed the activity by the text_block() is described.
[0121] title_info() is a field where the start point of title on chunkgroup, an ending point, and the attribute about other title(s) are written, as shown in drawing 16 . Moreover, title_info() can have flag which shows whether seamless playback can be guaranteed between titles, when it reproduces to a title numerical order. It is turned out whether whether the seamless playback of between titles can be carried out with an optical disk unit can grasp in advance, and it is necessary to change arrangement by this flag, at the time of a merge.

[0122] Since the boundary of a title is possible also on the boundary of file, seamless playback may not be guaranteed. However, it is possible as a function of an optical disk unit to change into the condition that seamless playback is generally performed by performing relocation etc.
[0123] title_info_length in title_info() of drawing 16 expresses the die length of title_info() per byte. level of the write-in attribute of corresponding title, writing (modification authorization), the count limit of playback, and rating etc. is recorded on flags_for_title. file_id of information file (CHUNKGROUP_###.CGIT) of chunkgroup which is base of corresponding title is recorded on cgit_file_id.
[0124] The time of day of the playback start point of the title on the local time-axis defined by chunkgroup is recorded on title_start_chunk_group_time_stamp. The display time of a picture which titleindex of that title has pointed out serves as this value. The time of day of the point of the title on the local time-axis defined by chunkgroup ending [playback] is recorded on title_end_chunk_group_stamp. This value becomes the same as that of the value which titleindex showing the start point of title located immediately after the point of chunkgroup ending [playback] or on a time-axis shows.
[0125] The playback time amount (a time code value, frame, or field number of sheets) of the title is recorded on title_playback_time(). The total (except for titleindex) of all mark(s) set up in the title is recorded on number_of_marks. As shown in drawing 37 , the class of mark attached to the location of the arbitration in title is recorded on mark_type. mark is used also as random access point in title. time stamp of the part where the mark is set up on the time-axis of chunkgroup is recorded on mark_chunk_group_time_stamp sequentially from what has a small value. Two or more mark(s) of the value with same time stamp must not exist. index which has the same time stamp as the start point of title and an ending point may exist. bytes which stuffing(s) is recorded on stuffing_bytes and the die length serves as 8xn bit (n>=0).
[0126] Next, chunkgroup and chunk which were shown in drawing 21 thru/or drawing 24 are explained further. CHUNKGROUP_###.CGIT is the file which described the definition of the time-axis of title and the configuration of chunk, and processing of the break point contained in title.
[0127] title(s) may be the case where it is stream which consists of various kinds of bitstream(s) and does not have video, and bitstream of DV (digital video). In a DV format, since formats differ when the time-axis is specified per frame and it is based on STC (System Time Clock) of MPEG 2 video, bitstream of this DV is not manageable.
[0128] Then, a local time-axis shall be set up within title. This time-axis is not dependent on stream which constitutes title. The boundary of title is set up regardless of the boundary of chunk. Therefore, it is more appropriate for a local time-axis every chunk (it corresponds to bitstream by 1 to 1) and to set up to the aggregate of chunk with which two or more title(s) (the number of arbitration) are contained rather than setting up for every title. The aggregate of this chunk is chunkgroup.
[0129] chunkgroup defined the single time-axis and the display time of day of chunk is set by sticking chunk on it. That is, chunk is located in a line with chunkgroup where the content (byte train) of bitstream file is developed on a time-axis. What arranged in on a time-axis all chunk(s) contained in one bitstream file is called path. Two or more path(s) can be arranged in chunkgroup. path which specifies the playback start time and end time of chunkgroup among path(s) is called main path, and other path(s) are called sub path. sub path mainly expresses chunk of audio by which additional record was carried out later etc.
[0130] Since the node of the boundary of title of chunk does not necessarily correspond, it is not

the attribute of title. However, about the relation between chunk(s), when it includes, there is a case which is the attribute of each chunk where conflict occurs hierarchical. It is thought appropriate for such break point information it is located in the medium of chunk and title and put on the hierarchy of chunkgroup.

[0131] When the above is summarized, the information which chunkgroup has is a break point which occurs in the method of the arrangement to up to the time-axis of chunk, the playback sequence of chunk, and the node of the end of chunk, and the beginning of chunk reproduced next.

[0132] As shown in drawing 38 , file_type_id of CHUNKGROUP_###.CGIT of drawing 21 is an identifier showing the file being CHUNKGROUP_###.CGIT, and is expressed with the character string of 16 characters according to ISO 646. flag about the criteria counter of chunkgroup is recorded on chunkgroup_time_base_flags. The start time of the conventional-time shaft in chunkgroup is recorded on chunkgroup_time_base_offset. This value is a value set to the counter which counts up a 90kHz clock, and is expressed with 32 bits. text_block() is a field for storing various text(s), and only text item allowed the activity by the text_block() is described.

[0133] As shown in drawing 22, chunk_connection_info() is a file for recording the information on a unique point (the changing point of video, synchronization of video and audio, etc.), and has specified the connection situation between chunk(s). It is necessary to change chunk in the middle of GOP in the singular point like the knot of chunk made by edit, and chunk. The information such near an editing point is described here. chunk does not belong to two or more chunkgroup(s).

[0134] What expressed the die length of chunk_connection_info() per byte is recorded on chunk_connection_info_length. The total of chunk used by the chunkgroup is recorded on number_of_chunks. chunk_sync_play_flag is flag which means whether it is necessary to reproduce two or more chunk(s) at this time of day as shown in drawing 39 , 0 of the value means playback of one chunk, and 1 of the value means simultaneous playback of two or more chunk(s).

[0135] In chunk_arrangement_info() of drawing 23 , what expressed the die length of the information about each chunk per byte (die length including from the head byte of chunk_arrangement_info_length to the last byte of transition_info()) is recorded on chunk_arrangement_info_length. file_id of target chunk_info_file is recorded on chunk_info_file_id.

[0136] When it connects, stream_id of stream which carries out continuation playback is recorded on chunk_switch_stream_id. As this id, id which identifies the video or the audio currently recorded on the packet header of MPEG 2 is used, for example. The time count value which expressed the display start time of Relevance chunk with the time of day in chunkgroup is recorded on presentation_start_cg_time_count. The display start time of chunk is expressed by global time stamp defined within chunkgroup. In chunkgroup, as for Relevance chunk, a display is started from this time of day. The time count value which expressed the display end time of Relevance chunk with the time of day in chunkgroup is recorded on presentation_end_cg_count. The display end time of chunk is expressed by global time stamp defined within chunkgroup.

[0137] As shown in drawing 40, the class of time count currently used inside stream is recorded on original_time_count_type. For example, original_time_count_type will be set to '0000' if it is stream of MPEG 2 video. When two or more time count is required, the number of time count which newly expresses required start time is recorded on number_of_start_original_time_count_extension. When two or more time count is required, the

number of time count which newly expresses required end time is recorded on number_of_end_original_time_count_extension. The time of day or the counter value in the interior of stream corresponding to presentation_start_tg_time_count is recorded on presentation_start_original_time_count. The time of day or the counter value in the interior of stream corresponding to presentation_end_tg_time_count is recorded on presentation_end_original_time_count.

[0138] The attribute over time_count_extension is recorded on tc_ext_attributes. The information on applying to which stream etc. can be put into this time_count_extension, for example. The start time required for a change or the initiation counter value of chunk is recorded on start_original_time_count_extension. This is an option, and when it is necessary to record two or more time of day and counter values, it is used. The end time required for a change or the termination counter value of chunk is recorded on end_original_time_count_extension. This is also an option, and it is used when it is necessary to record two or more time of day and counter values. When applying special effect by the change of chunk, required information is recorded on transition_info(). For example, the class of assignment of chunk, change time of day, and special effect etc. is described here.

[0139] As shown in drawing 24, CHUNK_%%%%.ABST is file which recorded the focus extracted from bitstream which constitutes sub_file No. No. %%%% chunk. That initiation byte location, die length, an attribute, etc. are described by this file for every unit which constitutes bitstream(s), such as GOP and Audio frame. GOP information and Audio frame information are summarized as one CHUNK_%%%%.ABST to every chunk (sub-file).

[0140] As shown in drawing 41, the identifier which shows that it is file on which stream_info() is recorded is recorded on file_type_id of CHUNK_%%%%.ABST by the character string of 16 characters according to ISO 646.

[0141] As shown in drawing 42 , type of stream_info which follows a degree in drawing 2424 is recorded on info_type. The class of stream is specified here. The number of program(s) contained in TS (Transport Stream) of MPEG 2 is recorded on number_of_programs. In order to get to know this number, it is necessary to read PSI (Program Specific Information). This value is set to 1 at the times other than TS. The number of the streams used by the program is recorded on number_of_streams. In the case of TS, this value becomes equal to the number of different PID (packet identification). In MPEGstream(s) other than TS, the number of stream(s) with which stream id differs is recorded here.

[0142] stream id is recorded on stream_identifier. In the case of TS, PID is used as stream id.

[0143] As shown in drawing 43, the way of dividing when dividing stream for every fixed spacing of a certain is recorded on slot_unit_type. When the index of breaks, such as each frame and field, is time amount, time stamp value is used. The time amount corresponding to 1slot is recorded on slot_time_length. This value is expressed with the value of time stamp using the counter which counts a 90kHz clock. The number of slot_info() currently written to CHUNK_%%%%.ABST is recorded on number_of_slots. The number of I-picture contained in slot is recorded on number_of_I_pictures_in_a_slot. This value turns into 15 or less value for one or more integers. However, the number of I-picture contained in slot located just before slot which makes GOPheader a head may be smaller than this value. This value is utilized when setting up slot which makes a head pictureheader of I-picture which is not immediately after GOPheader.

[0144] Next, program shown in drawing 17 and drawing 18 is explained further. Only one program() exists in PROGRAM_$$$.PGI. Only in the number of program(s),

PROGRAM_$$$.PGI exists in volume. A program number is not defined in program() but is prescribed by a file name or file id.
[0145] As shown in drawing 44 , id which shows that it is file on which program() was recorded is recorded on file_type_id of PROGRAM_$$$.PGI of drawing 17 by the character string of die length 16. The field for storing various text(s) is formed in text_block(). Here, only text item allowed the activity by the text_block() is described.
[0146] The various flags about program are recorded on flags_for_program of program() of drawing 18. For example, level of the write-in attribute of this program, writing (modification authorization), the count limit of playback, and rating etc. is recorded.
[0147] As shown in drawing 45 , the attribute of program is recorded on program_status. Although setting out of this field is option, when not setting up, it must be made into "none".
[0148] The playback time amount of the program is recorded on program_playback_time(). The number of play_sequence currently used by the program is recorded on number_of_play_sequences. However, the value is being fixed to 1 in the example of this format. Namely, what is necessary is just to enable simultaneous playback assignment of 2program in the example of this format, in order to realize 2ch simultaneous playback, since it considers as 1 program=1ch (channel) playback. If there is no limit of 1program=1ch playback, it is 1program and 2ch simultaneous playback is also possible. When carrying out simultaneous playback of two play sequence using multichannel I/O, an optical disk unit decides which play sequence to assign which output channel.
[0149] The number of play_list currently used by this play sequence is recorded on number_of_play_lists. A value is set to 1 in this example. The time of day within play sequence counted by timer started from the start time of play sequence is recorded on play_list_start_time_stamp_offset. This value becomes the start time of play list. In program, as for play list, only one must not exist in play sequence. The system of units of time of day is 90kHz (1 / 90000 seconds are the smallest units of time of day). bytes of stuffing is recorded on stuffing_bytes. The die length is set to 8xn bit (n>=0).
[0150] Next, a break point flag is explained further. A break point flag means the mark recorded as seamless_connection_flag of play_list() of drawing 19 , or indextype 8 shown in drawing 37 here.
[0151] 0 of the value of seamless_connection_flag means that continuation playback (seamless playback) with last play item is not guaranteed as shown in drawing 4646 , or an unknown thing, and 1 of the value means that seamless playback is guaranteed.
[0152] That is, as shown in drawing 47 , when this flag is 0, it is guaranteed that predetermined play item is continuously reproduced from last play item (without making an image or voice break off). On the other hand, when this flag is 1, after last play item is completed before following play item is reproduced, a discontinuous part may occur.
[0153] Next, with reference to the flow chart of drawing 48, the processing which records the break point flag of title is explained. First, in step S1, a user operates the input section 14 and specifies title made applicable to record. CPU21 judges [in / at this time / step S2] whether it is possible to reproduce that title continuously to title just before already being recorded, when title specified as the optical disk 1 at step S1 is recorded. This judgment is performed corresponding to the capacity (time amount which takes the data of that capacity for a decoder 7 to decode) of the buffer 6 for read-out channels, and the playback time amount between title(s). That is, when it is judged with continuation playback being impossible when the buffer 6 for read-out channels will become empty, by the time it reproduces the next title and does not become empty, it is

judged with continuation playback being possible.
[0154] When continuation playback is not possible, it progresses to step S3, CPU21 controls the OSD control circuit 9, and title ordered in record generates the message showing a break point occurring between the ability not to carry out continuation playback, i.e., both, to title already recorded now. This message is displayed on a display through an output terminal P1 from the synthetic circuit 8.
[0155] A user looks at this message, operates the input section 14 and inputs whether it consents to generating of a break point. When a user consents to generating of a break point, CPU21 performs processing which sets the location which generates a break point as relative_time_stamp_in_title of drawing 16 in step S5 while making index type 8 as an index which shows the break point of the marks shown in drawing 37 set it as mark_type in title_info() of drawing 16 . In addition, the location (location which generates a break point) recorded on this relative_time_stamp_in_title can be made into the location of arbitration. Therefore, even if a break point occurs, a point (for example, the head of the next title or the last of front title) with little effect is usually recorded here.
[0156] And title_info() to which such setting out was performed is written in from RAM24, is supplied to the buffer 11 for channels, and after memorizing, it is predetermined timing, and reading appearance of it is carried out from there, and it is supplied and recorded on an optical disk 1 through a switch 5, the ECC circuit 4, RF, the recovery/modulation circuit 3, and the optical head 2.
[0157] Next, it progresses to step S6, and CPU21 repeats and performs return and processing after it to step S3, when it judges whether the point which cannot follow others and cannot be reproduced exists and a playback break point otherwise exists in title. Processing is ended when judged with otherwise a playback break point not existing in title.
[0158] title on which it progresses to step S7 and CPU21, on the other hand, records the record location of title [finishing / record / already] after this when the purport which a user does not consent that generating of a break point is is inputted in step S4, and continuation -- it judges whether it can change into a refreshable location. title which records the record location of title which CPU21 progresses to step S8, and has already been recorded when it judges [that the record location of title can be changed, and] after this, and continuation -- it changes into a refreshable location. Then, it progresses to step S6.
[0159] In step S7, while a break point is recorded as mark_type as progressed and mentioned above to step S5 when judged with the ability of the record location of title not to be changed, the generating location of a break point is recorded on relative_time_stamp_in_title.
[0160] Record processing of title specified at step S1 is performed after such processing.
[0161] Next, with reference to the flow chart of drawing 49, the processing in the case of recording a break point flag on program creation time is explained. First, in step S21, a user operates the input section 14 and specifies title included in program. When this assignment is performed, in step S22, CPU21 specifies the required part in title specified at step S21 at a playback start point and the point ending [playback]. CPU21 creates play item (drawing 20) corresponding to this assignment.
[0162] In step S23 next, CPU21 It judges whether continuation playback is possible between last play item (this judgment with the playback time amount generated between play item). reading appearance is carried out and it is carried out based on the comparison with the capacity (decoding time amount of the capacity) of the buffer 6 for channels -- when continuation playback is not possible, it progresses to step S24, the OSD control circuit 9 is controlled, and it

is made to indicate that a break point occurs by the message A user inputs whether the input section 14 is operated and it consents to generating of a break point to this message. Then, in step S25, when it judges whether the user consented to generating of a break point and judges with the user having consented to generating of a break point, CPU21 progresses to step S26, and sets seamless_connection_flag of the play item as 0. As explained with reference to drawing 46 , that this flag is 0 means that continuation playback is not guaranteed.

[0163] Next, it progresses to step S27, it judges whether creation processing of program ended CPU21, and when it judges with having not ended yet, return and processing after it are repeated and performed to step S23. In step S27, when it judges with having ended creation processing of program, processing is ended.

[0164] On the other hand, when judged with a user not consenting to generating of a break point in step S25, it progresses to step S28, and CPU21 changes the record location on the optical disk 1 of the target stream (arrangement location), or judges whether partial re-record is possible. case relocation processing or partial re-record of a stream is possible -- step S29 -- progressing -- CPU21 -- a record location -- continuation -- processing changed into a refreshable location is performed. And in step S30, CPU21 sets seamless_connection_flag of that play item as 1 in this case. As explained with reference to drawing 46 , 1 of this flag means that continuation playback is guaranteed. Then, it progresses to step S27 and processing after it is performed.

[0165] In step S28, when judged with relocation processing or partial re-record of a stream being impossible, it progresses to step S26 and the same processing as the case where a user consents to generating of a break point is performed.

[0166] In step S23, when judged with continuation playback being possible between last play item, it progresses to step S30 and 1 is promptly set as seamless_connection_flag of the play item.

[0167] Next, as explained with reference to drawing 48, when a break point flag is recorded on title, the processing when being ordered in the playback is explained with reference to the flow chart of drawing 50 . First, in step S41, CPU21 reads the break point flag of specified title. In step S42, it judges whether title which started regeneration of title and started playback in step S43 ended CPU21. When playback of title is not completed, it progresses to step S44, and it judges, and when judged with not being reproduced yet, it returns [whether the location where CPU21 was expressed with the break point flag (relative_time_stamp_in_title) was reproduced, and] to step S43.

[0168] In step S44, when judged with the location expressed with the break point flag having been reproduced, it progresses to step S45, and since a player (in the case of now optical disk unit) is continuation playback of a consecutiveness part, CPU21 judges whether a gap is required. This judgment is performed to the buffer 6 for read-out channels based on the amount of data (time amount which takes that amount of data for a decoder 7 to decode) memorized at that event. The amount of data comes out enough, and, in a certain case, it is judged with a gap being unnecessary, and when the amount of data is insufficient, it is judged with a gap being required. Since this optical disk unit is continuation playback of a consecutiveness part, when you do not need the gap, return and processing after it are repeatedly performed by step S43 (when data are enough memorized by the buffer 6 for read-out channels).

[0169] On the other hand, in step S45, when judged with needing the gap for playback of an optical disk unit of a consecutiveness part, it progresses to step S46 (when data are not enough memorized by the buffer 6 for read-out channels), and CPU21 generates a gap. That is, although the writing of data to the buffer 6 for read-out channels makes CPU21 continue as it is, you make

it interrupted and the read-out to a decoder 7 makes the amount of data of the buffer 6 for read-out channels increase. And when it progresses to step S47, it judges whether continuation playback of a consecutiveness part was attained and it is not possible yet, step S46 is made to continue return and gap generating processing.

[0170] The data of the specified quantity read as mentioned above, and it is written in the buffer 6 for channels, and when judged with having changed into the condition in which continuation playback of a consecutiveness part is possible in step S47, it progresses to step S48 and CPU21 makes playback continue again (decoding by the decoder 7 is resumed). Then, return and processing after it are repeatedly performed by step S43.

[0171] On the other hand, when judged with title having been completed in step S43, it progresses to step S49 and, as for CPU21, judges whether title reproduced next exists. Next, processing is ended when title to reproduce does not exist. On the other hand, when judged with title reproduced next existing, it progresses to step S50 and CPU21 judges whether a discontinuous flag exists in the last of front title. When judged with a discontinuous flag existing in the last of title reproduced before, it progresses to step S51 and CPU21 judges whether a player (optical disk unit) needs a gap for playback of the next title. When judged with a gap being required, it progresses to step S52, CPU21 performs gap generating processing, and in step S53, when playback of the next title judges whether it became possible and is not possible, return and gap generating processing are repeated and performed to step S52.

[0172] In step S53, when judged with playback of the next title having been attained, return and processing after it are repeatedly performed by step S41.

[0173] In step S50, when judged with a discontinuous flag not existing in the last of front title, return and processing after it are performed by step S41. Moreover, even if a discontinuous flag exists in the last of front title, when it is judged with not needing a gap for playback of an optical disk unit of the next title in step S51, return and processing after it are performed by step S41 (when the capacity of the buffer 6 for read-out channels of the optical disk unit is sufficiently large).

[0174] Next, as it was shown in drawing 49, when program is created, the processing when being ordered in playback of the program is explained with reference to the flow chart of drawing 51 .

[0175] First, in step S61, CPU21 reproduces the first play item which constitutes specified program. In step S62, when judging whether playback of play item ended CPU21 and having not ended yet, it stands by until it ends.

[0176] In step S62, when judged with playback of play item having been completed, CPU21 progresses to step S63, and judges whether following play item exists. When judged with following play item existing, it progresses to step S64 and CPU21 judges whether seamless_connection_flag of following play item is 0. When it judges whether it progresses to step S65 (when seamless playback is not guaranteed), and CPU21 needs a gap for the next play item playback of an optical disk unit, when judged with this flag being 0 and you need a gap, it progresses to step S66 and performs gap generating processing. And in step S67, when it judges whether it changed into the condition which can reproduce following play item and is not in the still possible condition, return and gap generating processing are repeated and performed to step S66.

[0177] In step S67, when judged with having changed into the condition which can reproduce following play item, it progresses to step S68 (when the data of sufficient amount for the buffer 6 for read-out channels are written in), and CPU21 starts regeneration of following play item. Then, return and processing after it are repeatedly performed by step S62.

[0178] In step S64, when judged with seamless_connection_flag of following play item not being 0, or when it is judged with not needing the gap in step S65 again for the next play item playback of an optical disk unit (when judged with it being 1), processing of step S66 and step S67 is skipped, and progresses to step S68. And playback of following play item is started and it returns to step S62 after that.
[0179] Although the case where this invention was applied to an optical disk unit was explained above as an example, this invention can be applied also when recording or reproducing information to other record media.
[0180] In addition, as an offer medium which provides a user with the computer program which performs processing which was described above, communication media, such as a network besides record media, such as a magnetic disk, CD-ROM, and solid-state memory, and a satellite, can be used.
[0181]
[Effect of the Invention] Since it judges whether it is continuously refreshable to the data already recorded in data and was made to perform processing about continuation playback like the above corresponding to the judgment result according to the record regenerative apparatus according to claim 1, the record playback approach according to claim 4, and the offer medium according to claim 5, the record medium which can secure compatibility is realizable.
[0182] Since the gap was added corresponding to the continuation playback information extracted from the record medium according to the record regenerative apparatus according to claim 6, the record playback approach according to claim 7, and the offer medium according to claim 8, compatibility is securable irrespective of the difference in the capacity of the buffer of a record regenerative apparatus. Therefore, it can control that a user takes for failure of equipment.

[Translation done.]

(2)

(19)日本国特許庁（ＪＰ）　(12)公開特許公報（A）　(11)特許出願公開番号

特開平11－317014

(43)公開日　平成11年(1999)11月16日

(51)Int.Cl.⁶		識別記号	FI		
G11B	20/10	301	G11B	20/10	301Z
	19/02	501		19/02	501J

審査請求　未請求　請求項の数8　ＯＬ　（全 34 頁）

(21)出願番号　特願平10－120393

(22)出願日　平成10年(1998)４月30日

(71)出願人　000002185
ソニー株式会社
東京都品川区北品川６丁目７番35号

(72)発明者　藤波　靖
東京都品川区北品川６丁目７番35号　ソニー株式会社内

(72)発明者　浜田　俊也
東京都品川区北品川６丁目７番35号　ソニー株式会社内

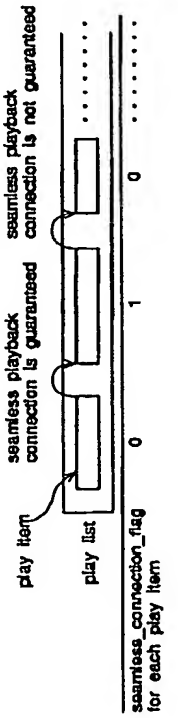
(74)代理人　弁理士　稲本　義雄

(54)【発明の名称】　記録再生装置および方法、並びに提供媒体

(57)【要約】

【課題】　装置の互換性を確保する。

【解決手段】　直前のplay itemに続いて、次のplay itemをシームレスに再生することができるとき、seamless_connection_flagを１に設定し、シームレス再生が保証できないとき、０に設定する。



【特許請求の範囲】
【請求項１】　記録媒体に対してデータを記録または再生する記録再生装置において、
前記データを前記記録媒体に対して記録する記録手段と、
前記データを前記記録媒体に記録したとき、そのデータを、既に記録されているデータに対して連続して再生可能であるか否かを判定する判定手段と、
前記判定手段の判定結果に対応して、データの連続再生に関する処理を実行する実行手段とを備えることを特徴とする記録再生装置。
【請求項２】　前記実行手段は、前記判定手段が、前記連続再生が不可能であると判定したとき、前記データの記録位置を変更することを特徴とする請求項１に記載の記録再生装置。
【請求項３】　前記実行手段は、前記判定手段が、前記連続再生が不可能であると判定したとき、連続再生が不可能であることを表す連続再生情報を前記記録媒体に記録することを特徴とする請求項１に記載の記録再生装置。
【請求項４】　記録媒体に対してデータを記録または再生する記録再生装置の記録再生方法において、
前記データを前記記録媒体に対して記録する記録ステップと、
前記データを前記記録媒体に記録したとき、そのデータを、既に記録されているデータに対して連続して再生可能であるか否かを判定する判定ステップと、
前記判定ステップでの判定結果に対応して、データの連続再生に関する処理を実行する実行ステップとを含むことを特徴とする記録再生方法。
【請求項５】　記録媒体に対してデータを記録または再生する記録再生装置に、
前記データを前記記録媒体に対して記録する記録ステップと、
前記データを前記記録媒体に記録したとき、そのデータを、既に記録されているデータに対して連続して再生可能であるか否かを判定する判定ステップと、
前記判定ステップでの判定結果に対応して、データの連続再生に関する処理を実行する実行ステップとを含む処理を実行させるプログラムを提供することを特徴とする提供媒体。
【請求項６】　記録媒体に対してデータを記録または再生する記録再生装置において、
前記記録媒体に記録されているデータを再生する再生手段と、
前記再生手段により再生されたデータから、前記データの連続再生が可能か否かを表す連続再生情報を抽出する抽出手段と、
前記抽出手段により抽出された前記連続再生情報に対応して、前記再生手段により再生されたデータに対してギャップを付加する付加手段とを備えることを特徴とする記録再生装置。
【請求項７】　記録媒体に対してデータを記録または再生する記録再生装置の記録再生方法において、
前記記録媒体に記録されているデータを再生する再生ステップと、
前記再生ステップで再生されたデータから、前記データの連続再生が可能か否かを表す連続再生情報を抽出する抽出ステップと、
前記抽出ステップで抽出された前記連続再生情報に対応して、前記再生ステップで再生されたデータに対してギャップを付加する付加ステップとを含むことを特徴とする記録再生方法。
【請求項８】　記録媒体に対してデータを記録または再生する記録再生装置に、
前記記録媒体に記録されているデータを再生する再生ステップと、
前記再生ステップで再生されたデータから、前記データの連続再生が可能か否かを表す連続再生情報を抽出する抽出ステップと、
前記抽出ステップで抽出された前記連続再生情報に対応して、前記再生ステップで再生されたデータに対してギャップを付加する付加ステップとを含む処理を実行させるプログラムを提供することを特徴とする提供媒体。
【発明の詳細な説明】
【０００１】
【発明の属する技術分野】本発明は、記録再生装置および方法、並びに提供媒体に関し、特に、データの連続再生に関連して、ユーザが故障と誤認するようなことを抑制するようにした記録再生装置および方法、並びに提供媒体に関する。
【０００２】
【従来の技術】ディスクには、ビデオデータやオーディオデータなどが連続して記録されるだけでなく、断続的に時間をおいて記録されることがある。また、一旦記録されたデータが消去され、他のデータが上書きされる場合がある。
【０００３】
【発明が解決しようとする課題】このような消去あるいは上書き処理を繰り返し実行すると、連続して再生すべきデータが必ずしもディスク上の連続する位置に記録されず、ディスク上の離間した位置に記録される場合がある。このような場合に、装置の再生時のバッファの容量が充分ではないと、その記録位置によっては、データを連続的に再生することができず、再生データが一時的に欠落するような場合がある。
【０００４】しかしながら、不連続部分が発生するか否かは、装置のバッファの容量に影響されるので、装置間の互換性を確保することができず、最悪の場合、ユーザが、装置が故障したと誤認するおそれがあった。

【０００５】本発明はこのような状況に鑑みてなされたものであり、装置の互換性を確保し、ユーザが装置が故障したと誤認するのを抑制するようにするものである。

【０００６】

【課題を解決するための手段】請求項１に記載の記録再生装置は、データを記録媒体に対して記録する記録手段と、データを記録媒体に記録したとき、そのデータを、既に記録されているデータに対して連続して再生可能であるか否かを判定する判定手段と、判定手段の判定結果に対応して、データの連続再生に関する処理を実行する実行手段とを備えることを特徴とする。

【０００７】請求項４に記載の記録再生方法は、データを記録媒体に対して記録する記録ステップと、データを記録媒体に記録したとき、そのデータを、既に記録されているデータに対して連続して再生可能であるか否かを判定する判定ステップと、判定ステップでの判定結果に対応して、データの連続再生に関する処理を実行する実行ステップとを含むことを特徴とする。

【０００８】請求項５に記載の提供媒体は、記録媒体に対してデータを記録または再生する記録再生装置に、データを記録媒体に対して記録する記録ステップと、データを記録媒体に記録したとき、そのデータを、既に記録されているデータに対して連続して再生可能であるか否かを判定する判定ステップと、判定ステップでの判定結果に対応して、データの連続再生に関する処理を実行する実行ステップとを含む処理を実行させるプログラムを提供することを特徴とする。

【０００９】請求項６に記載の記録再生装置は、記録媒体に記録されているデータを再生する再生手段と、再生手段により再生されたデータから、データの連続再生が可能か否かを表す連続再生情報を抽出する抽出手段と、抽出手段により抽出された連続再生情報に対応して、再生手段により再生されたデータに対してギャップを付加する付加手段とを備えることを特徴とする。

【００１０】請求項７に記載の記録再生方法は、記録媒体に記録されているデータを再生する再生ステップと、再生ステップで再生されたデータから、データの連続再生が可能か否かを表す連続再生情報を抽出する抽出ステップと、抽出ステップで抽出された連続再生情報に対応して、再生ステップで再生されたデータに対してギャップを付加する付加ステップとを含むことを特徴とする。

【００１１】請求項８に記載の提供媒体は、記録媒体に対してデータを記録または再生する記録再生装置に、記録媒体に記録されているデータを再生する再生ステップと、再生ステップで再生されたデータから、データの連続再生が可能か否かを表す連続再生情報を抽出する抽出ステップと、抽出ステップで抽出された連続再生情報に対応して、再生ステップで再生されたデータに対してギャップを付加する付加ステップとを含む処理を実行させるプログラムを提供することを特徴とする。

【００１２】請求項１に記載の記録再生装置、請求項４に記載の記録再生方法、および請求項５に記載の提供媒体においては、既に記録されているデータに対して連続して再生可能であるか否かの判定結果に対応して、データの連続再生に関する処理が実行される。

【００１３】請求項６に記載の記録再生装置、請求項７に記載の記録再生方法、および請求項８に記載の提供媒体においては、記録媒体から再生された連続再生情報に対応して、データに対してギャップが付加される。

【００１４】

【発明の実施の形態】以下に本発明の実施の形態を説明するが、特許請求の範囲に記載の発明の各手段と以下の実施の形態との対応関係を明らかにするために、各手段の後の括弧内に、対応する実施の形態（但し一例）を付加して本発明の特徴を記述すると、次のようになる。但し勿論この記載は、各手段を記載したものに限定することを意味するものではない。

【００１５】請求項１に記載の記録再生装置は、データを記録媒体に対して記録する記録手段（例えば、図２５の光ヘッド２）と、データを記録媒体に記録したとき、そのデータを、既に記録されているデータに対して連続して再生可能であるか否かを判定する判定手段（例えば、図４８のステップＳ２）と、判定手段の判定結果に対応して、データの連続再生に関する処理を実行する実行手段（例えば、図４８のステップＳ５）とを備えることを特徴とする。

【００１６】請求項６に記載の記録再生装置は、記録媒体に記録されているデータを再生する再生手段（例えば、図２５の光ヘッド２）と、再生手段により再生されたデータから、データの連続再生が可能か否かを表す連続再生情報を抽出する抽出手段（例えば、図５０のステップＳ４１）と、抽出手段により抽出された連続再生情報に対応して、再生手段により再生されたデータに対してギャップを付加する付加手段（例えば、図５０のステップＳ４６）とを備えることを特徴とする。

【００１７】最初に本発明において情報が記録または再生される記録媒体（メディア）上のファイル配置について説明する。メディア上には、図１に示すように、次の７種類のファイルが記録される。

VOLUME.TOC
ALBUM.STR
PROGRAM_$$$.PGI
TITLE_###.VDR
CHUNKGROUP_@@@.CGIT
CHUNK_%%%%.ABST
CHUNK_%%%%.MPEG2

【００１８】ルートディレクトリにはVOLUME.TOCおよびALBUM.STRが置かれる。また、ルートディレクトリ直下のディレクトリ"PROGRAM"には、"PROGRAM_$$$.PGI"（ここで"$$$"はプログラム番号を表す）が置かれる。同様

に、ルートディレクトリ直下のディレクトリ"TITLE"には、"TITLE_###.VDR"（ここで"###"はタイトル番号を表す）が、ディレクトリ"CHUNKGROUP"には、"CHUNKGROUP_@@@.CGIT"（ここで"@@@"はチャンクグループ番号を表す）が、ディレクトリ"CHUNK"には、"CHUNK_%%%%.ABST"（ここで"%%%%"はチャンク番号を表す）が、それぞれ置かれる。

【００１９】ルートディレクトリ直下のMPEGAVディレクトリには、更に１つ以上のサブディレクトリが作成され、その下に、"CHUNK_%%%%.MPEG2"（ここで%%%%はチャンク番号を表す）が置かれる。

【００２０】VOLUME.TOCのファイルは、メディア上に１つ有るのが普通である。ただし、ROMとRAMのハイブリッド構造のメディア等、特殊な構造のメディアでは、複数存在することも有り得る。このファイルは、メディアの全体の性質を示すために用いられる。

【００２１】VOLUME.TOCの構造は図２に示すようになっている。先頭にfile_type_idが置かれ、これにより該当ファイルがVOLUME.TOCであることが示される。次にvolume_information()が続き、最後にtext_block()が続く。

【００２２】図３にvolume_information()の構成が示されている。これは、volume_attribute()、resume()、volume_rating()、write_protect()、play_protect()、recording_timer()を含んでいる。

【００２３】volume_attribute()は、logical volumeの属性を記録する領域であり、図４にその詳細な構造が示されている。同図に示すように、この領域には、title_playback_mode_flag, program_playback_mode_flagなどが含まれている。

【００２４】resume()は、メディアの再挿入時に、eject直前の状態を復元するための情報を記録する領域であり、その詳細な構造は、図５に示されている。

【００２５】図３のvolume_rating()は、volume全体に対する視聴年齢制限を年齢やカテゴリに応じて実現するための情報を記録する領域であり、その詳細な構造は、図６に示されている。

【００２６】図３のwrite_protect()は、volume内に記録されているtitle, programに対する変更や、消去操作を制限する情報を記録する領域であり、その詳細な構造は、図７に示されている。

【００２７】図３のplay_protect()は、volume内に記録されているtitle, programに対する再生許可、不許可の設定、あるいは、再生回数を制限する情報を記録する領域であり、その詳細な構造は、図８に示されている。

【００２８】図３のrecording_timer()は、記録時間を制御する情報を記録する領域であり、その詳細な構造は、図９に示されている。

【００２９】図２のVOLUME.TOCのtext_block()の詳細な構造は図１０に示されている。このtext_block()には、language_set()とtext_itemが含まれており、その詳細な構造は図１１と図１２にそれぞれ示されている。

【００３０】図１のALBUM.STRのファイルは、メディア上に１つ有るのが普通である。ただし、ROMとRAMのハイブリッド構造のメディア等、特殊な構造のメディアでは、複数存在することも有り得る。このファイルは、複数のメディアを組み合わせて、あたかも１つのメディアであるような構成にするために使用される。

【００３１】このALBUM.STRの構造は、図１３に示すようになっている。先頭にfile_type_idが置かれ、該当ファイルがALBUM.STRであることを示す。次にalbum()が続き、最後にtext_block()が続く。

【００３２】album()は、複数のvolume（複数のメディア）を１つのまとまりとして扱うための情報を記録する領域であり、その詳細な構造は、図１４に示されている。

【００３３】図１のTITLE_###.VDRのファイルは、タイトルの数だけ存在する。タイトルとは、例えばcompact discで言うところの１曲や、テレビ放送の１番組を言う。この情報の構造は図１５に示すようになっている。先頭にfile_type_idが置かれ、これにより該当ファイルがTITLE_###.VDRであることが示される。次にtitle_info()が続き、最後にtext_block()が続く。###はタイトル番号を示す文字列である。

【００３４】title_info()は、chunkgroup上における、titleの開始点、終了点、その他titleに関する属性を記録するための領域であり、その詳細な構造は、図１６に示されている。

【００３５】図１のPROGRAM_$$$.PGIのファイルは、プログラムの数だけ存在する。プログラムは、タイトルの一部（あるいは全部）の領域を指定した複数のカットで構成され、各カットは指定された順番で再生される。この情報の構造は図１７に示されている。先頭にfile_type_idが置かれ、該当ファイルがPROGRAM_$$$.PGIであることを示す。次にprogram()が続き、最後にtext_block()が続く。$$$はタイトル番号を示す文字列である。

【００３６】program()は、素材に対して不可逆な編集を施すことなしに、titleの必要な部分を集めて再生するのに必要な情報を記録する領域であり、その詳細な構造は、図１８に示されている。

【００３７】図１８のprogram()は、１つのplay_listを有している。このplay_list()の詳細は、図１９に示されている。

【００３８】play_listには、play_item()が複数置かれている。play_item()の詳細は、図２０に示されている。

【００３９】図１のCHUNKGROUP_@@@.CGITのファイルは、チャンクグループの数だけ存在する。チャンググループはビットストリームを並べるためのデータ構造である。このファイルは、ユーザがVDR（ビデオディスクレコーダ）など、メディアを記録再生する装置を普通に操

作している分にはユーザに認識されない。

【００４０】この情報の構造は図２１に示すようになっている。先頭にfile_type_idが置かれ、該当ファイルがCHUNKGROUP_000.CGITであることを示す。その次にchunkgroup_time_base_flagsとchunkgroup_time_base_offsetが有り、次にchunk_connection_info()、最後にtext_block()が続く。

【００４１】chunkgroup_time_base_flagsは、chunkgroupの基準カウンタに関するflagを示し、chunkgroup_time_base_offsetは、chunkgroup内の基準時間軸の開始時刻を示す。これは、９０kHzでカウントアップするカウンタにセットする値であり、３２ビットの大きさを有する。chunk_connection_info()は、videoの切換点や、videoとaudioの同期など、特異な点の情報を記憶する領域であり、その詳細な構造は、図２２に示されている。

【００４２】このchunk_connection_info()には、チャンクグループに属するチャンクの数だけchunk_arrangement_info()のループが置かれる。図２３にこのchunk_arrangement_info()の詳細が示されている。

【００４３】図１のCHUNK_%%%%.ABSTのファイルは、チャンクの数だけ存在する。チャンクはストリームファイル１つに対応する情報ファイルである。この情報の構造は図２４に示すようになっている。先頭にfile_type_idが置かれ、これにより、該当ファイルがCHUNK_%%%%.ABSTであることが示される。

【００４４】図１のCHUNK_%%%%.MPEG2のファイルは、ストリームファイルである。このファイルはMPEGのビットストリームを格納しており、この他のファイルが情報のみを記録しているのと異なっている。

【００４５】図２５は、以上のようなファイルを有するメディアとしての光ディスクに対して情報を記録または再生する光ディスク装置の構成例を表している。この光ディスク装置では、１枚の書き換え型の光ディスク１に対して１系統の光ヘッド２が設けられており、データの読み出しと書き込みの双方にこの光ヘッド２が共用される。

【００４６】光ヘッド２により光ディスク１から読み出されたビットストリームは、ＲＦおよび復調／変調回路３で復調された後、ECC回路４で誤り訂正が施され、スイッチ５を介して、読み出しレートとデコード処理レートとの差を吸収するための読み出しチャネル用バッファ６に送られる。読み出しチャネル用バッファ６の出力はデコード７に供給されている。読み出しチャネル用バッファ６はシステムコントローラ１３から読み書きができるように構成されている。

【００４７】読み出しチャネル用バッファ６から出力されたビットストリームは、デコーダ７でデコードされ、そこからビデオ信号とオーディオ信号が出力される。デコーダ７から出力されたビデオ信号は合成回路８に入力され、OSD (On Screen Display) 制御回路９が出力するビデオ信号と合成されたた後、出力端子Ｐ１から図示せぬディスプレイに出力され、表示される。デコーダ７から出力されたオーディオ信号は、出力端子Ｐ２から図示せぬスピーカに送られて再生される。

【００４８】他方、入力端子Ｐ３から入力されたビデオ信号、および入力端子Ｐ４から入力されたオーディオ信号は、エンコーダ１０でエンコードされた後、エンコード処理レートと書き込みレートとの差を吸収するための書き込みチャネル用バッファ１１に送られる。この書き込みチャネル用バッファ１１もシステムコントローラ１３から読み書きができるように構成されている。

【００４９】書き込みチャネル用バッファ１１に蓄積されたデータは、書き込みチャネル用バッファ１１から読み出され、スイッチ５を介してECC回路４に入力されて誤り訂正符号が付加された後、ＲＦおよび復調／変調回路３で変調される。ＲＦおよび復調／変調回路３より出力された信号（ＲＦ信号）は、光ヘッド２により光ディスク１に書き込まれる。

【００５０】アドレス検出回路１２は、光ディスク１の記録または再生するトラックのアドレス情報を検出する。システムコントローラ１３は、この光ディスク装置の各部の動作を制御するものであり、各種の制御を行うCPU２１、CPU２１が実行すべき処理プログラム等を格納したROM２２、処理過程で生じたデータ等を一時記憶するためのRAM２３、および光ディスク１に対して記録または再生する各種の情報ファイルを記憶するRAM２４を有している。CPU２１は、アドレス検出回路１２の検出結果に基づいて、光ヘッド２の位置を微調整する。CPU２１はまた、スイッチ５の切り替え制御を行う。各種のスイッチ、ボタンなどから構成される入力部１４は、各種の指令を入力するとき、ユーザにより操作される。

【００５１】次に、基本的な情報ファイルの読み込み動作について説明する。例えば、"VOLUME.TOC"情報ファイルの読み込みを行うとき、ステムコントローラ１３のCPU２１は、予めその処理プログラムに組み込んであるファイルシステム操作命令を使用し、"VOLUME.TOC"が記録されている光ディスク１上の物理アドレスと、その長さを確定する。続いて、CPU２１は、この"VOLUME.TOC"のアドレス情報に基づき、光ヘッド２を読み出し位置に移動させる。そしてCPU２１は、光ヘッド２、ＲＦおよび復調／変調回路３、並びにECC回路４を読み出しモードに設定するとともに、スイッチ５を読み出しチャネル用バッファ６側に切り替え、さらに光ヘッド２の位置を微調整した後、光ヘッド２による読み出しを開始させる。これにより"VOLUME.TOC"の内容が光ヘッド２により読み出され、ＲＦおよび復調／変調回路３により復調され、さらにECC回路４により誤り訂正が行われた後、読み出しチャネル用バッファ６に蓄積される。

【００５２】読み出しチャネル用バッファ６に蓄積されたデータ量が、"VOLUME.TOC"の大きさと等しいか、ある

いはより大きくなった時点で、CPU２１は読み出しを停止させる。その後、CPU２１は、読み出しチャネル用バッファ６から該当データを読み出し、RAM２４に記憶させる。

【００５３】次に、基本的な情報ファイル書込み動作について、"VOLUME.TOC"情報ファイルを書き込む場合を例として説明する。CPU２１は、予めその処理プログラムに組み込んであるファイルシステム操作命令を使用し、ファイルシステム（光ディスク１）中に、これから書こうとしている"VOLUME.TOC"と等しいか、より大きい大きさを持つ空き領域を探し、そのアドレスを確定する。

【００５４】次に、CPU２１は、RAM２４に用意されている、新たに書き込むべき"VOLUME.TOC"を、書き込みチャネル用バッファ１１に転送する。続いて、CPU２１は、空き領域のアドレス情報に基づき、光ヘッド２を書き込み位置に移動させる。そしてCPU２１は、光ヘッド２、ＲＦおよび復調／変調回路３、並びにECC回路４を書き込みモードに設定するとともに、スイッチ５を書き込みチャネル用バッファ１１側に切り替え、光ヘッド２の位置を微調整した後、光ヘッド２による書き込みを開始させる。

【００５５】これにより新たに用意した"VOLUME.TOC"の内容が、書き込みチャネル用バッファ１１から読み出され、スイッチ５を介してECC回路４に入力され、誤り訂正符号が付加された後、ＲＦおよび復調／変調回路３により変調される。ＲＦおよび復調／変調回路３より出力された信号は、光ヘッド２により光ディスク１に記録される。書き込みチャネル用バッファ１１から読み出され、光ディスク１に記録されたデータ量が、"VOLUME.TOC"の大きさと等しくなった時点で、CPU２１は書き込み動作を停止させる。

【００５６】最後に、CPU２１は、予めその処理プログラムに組み込んであるファイルシステム操作命令を使用し、ファイルシステム（光ディスク１）中の"VOLUME.TOC"を指し示すポインタを、新しく書込んだ位置を指し示すように書き換える。

【００５７】次に、基本的なストリーム再生動作について、図１のCHUNK_0001.MPEG2というストリームを再生する場合を例として説明する。CPU２１は、予めその処理プログラムに組み込んであるファイルシステム操作命令を使用し、"CHUNK_0001.MPEG2"が記録されている光ディスク１上の物理アドレスと、その長さを確定する。続いて、CPU２１は、この"CHUNK_0001.MPEG2"のアドレス情報に基づき、光ヘッド２を読み出し位置に移動させる。そして光ヘッド２、ＲＦおよび復調／変調回路３、並びにECC回路４を読み出しモードに設定するとともに、スイッチ５を読み出しチャネル用バッファ６側に切り替え、光ヘッド２の位置を微調整した後、光ヘッド２による読み出しを開始させる。

【００５８】光ヘッド２により読み出された"CHUNK_0001.MPEG2"の内容が、ＲＦおよび復調／変調回路３、ECC回路４、並びにスイッチ５を介して読み出しチャネル用バッファ６に蓄積される。読み出しチャネル用バッファ６に蓄積されたデータは、デコーダ７に出力され、デコード処理が施されて、ビデオ信号とオーディオ信号がそれぞれ出力される。オーディオ信号は出力端子Ｐ２から出力され、ビデオ信号は、合成回路８を介して出力端子Ｐ１から出力される。

【００５９】光ディスク１から読みだされ、デコード、表示されたデータ量が、"CHUNK_0001.MPEG2"の大きさと等しくなった時点で、あるいは、入力部１４から読み出し動作の停止が指定された時点で、CPU２１は、読み出しおよびデコード処理を停止させる。

【００６０】次に、基本的なストリーム記録動作を、"CHUNK_0001.MPEG2"情報ファイルを書き込む場合を例として説明する。CPU２１は、予めその処理プログラムに組み込んであるファイルシステム操作命令を使用し、ファイルシステム（光ディスク１）中にこれから書こうとしている"CHUNK_0001.MPEG2"と等しいか、それより大きい大きさを持つ空き領域を探し、そのアドレスを確定する。

【００６１】入力端子Ｐ３から入力されたビデオ信号、および入力端子Ｐ４から入力されたオーディオ信号は、エンコーダ１０によりエンコードされた後、書き込みチャネル用バッファ１１に蓄積される。続いて、CPU２１は、空き領域のアドレス情報に基づき、光ヘッド２を書き込み位置に移動させる。そしてCPU２１は、光ヘッド２、ＲＦおよび復調／変調回路３、並びにECC回路４を書き込みモードに設定するとともに、スイッチ５を書き込みチャネル用バッファ１１側に切り替え、光ヘッド２の位置を微調整した後、光ヘッド２による書き込みを開始させる。これにより新たに用意した"CHUNK_0001.MPEG2"の内容が、書き込みチャネル用バッファ１１から読み出され、スイッチ５、ECC回路４、ＲＦおよび復調／変調回路３を介して光ヘッド２に入力され、光ディスク１に記録される。

【００６２】書き込みチャネル用バッファ１１から読み出され、光ディスク１に記録されたデータ量が、予め設定した値と等しくなったとき、あるいは入力部１４から書き込み動作の停止が指定されたとき、CPU２１は書き込み動作を停止させる。最後に、CPU２１は、予めその処理プログラムに組み込んであるファイルシステム操作命令を使用し、ファイルシステム（光ディスク１）中の"CHUNK_0001.MPEG2"を指し示すポインタを、新しく書込んだ位置を指し示すように書き換える。

【００６３】いま、光ディスク１に、図２６に示すような情報ファイルとストリームファイルが記録されているものとする。この例では、"PROGRAM_001.PGI"という名前の１つのプログラムのファイルが含まれている。また、この光ディスク１には、"TITLE_001.VDR"、"TITLE_

002.VDR"、および"TITLE_003.VDR"という名前の３つのタイトルのファイルが含まれている。

【００６４】さらに、この光ディスク１には、"CHUNKGROUP_001.CGIT"と"CHUNKGROUP_002.CGIT"という２つのチャンクグループのファイルが含まれている。また、この光ディスク１には、"CHUNK_0001.MPEG2"、"CHUNK_0011.MPEG2"、および"CHUNK_0012.MPEG2"という名前の３つのストリームのファイルが含まれているとともに、それぞれに対応する情報として、"CHUNK_0001.ABST"、"CHUNK_0011.ABST"、および"CHUNK_0012.ABST"の３つの情報ファイルが置かれている。

【００６５】図２６に示した情報ファイルとストリームファイルを有する光ディスク１の論理構造は、図２７に示すようになる。この例では、チャンク情報ファイル"CHUNK_0001.ABST"は、ストリームファイル"CHUNK_0001.MPEG2"を、またチャンク情報ファイル"CHUNK_0011.ABST"は、ストリームファイル"CHUNK_0011.MPEG2"を、さらに、チャンク情報ファイル"CHUNK_0012.ABST"は、ストリームファイル"CHUNK_0012.MPEG2"を、それぞれ指定している。具体的には、図２４のCHUNK_%%%%.ABST中の、chunk_file_idというフィールドで、ストリームのファイルIDが指定される。

【００６６】さらに、この例では、チャンクグループ情報ファイル"CHUNKGROUP_001.CGIT"は、チャンク情報ファイル"CHUNK_0001.ABST"を、またチャンクグループ情報ファイル"CHUNKGROUP_002.CGIT"は、チャンク情報ファイル"CHUNK_0011.ABST"と"CHUNK_0012.ABST"を、それぞれ指定している。具体的には、図２３のchunk_arrangement_info()の中のchunk_info_file_idというフィールドでチャンク情報のファイルIDが指定される。このchunk_arrangement_info()はチャンクグループ情報ファイルの中にあり、該当チャンクグループに属するチャンクの数だけ存在するデータ構造となっている（図２３のchunk_arrangement_info()は、図２２のchunk_connection_info()に記述されており、このchunk_connection_info()は、図２１のCHUNKGROUP_###.CGITに記述されている）。

【００６７】CHUNKGROUP_001には、chunk_arrangement_info()が１つだけあり、その中のchunk_info_file_idがCHUNK_0001を指定している。CHUNKGROUP_002には、chunk_arrangement_info()が２つあり、その中で、それぞれCHUNK_0011とCHUNK_0012が指定されている。このような場合のため、チャンクグループは、複数のチャンクの再生順序等を指定できるようになっている。

【００６８】具体的には、まず、図２１のCHUNKGROUP_###.CGIT中のchunkgroup_time_base_offsetにより、該当チャンクグループでの時計の初期値が定められる。次に各チャンクを登録する際に、図２３のchunk_arrangement_info()のpresentation_start_cg_countとpresentation_end_cg_time_countが指定される。

【００６９】例えば、図２８に示すように、CHUNK_0011の長さ（時間）をA、CHUNK_0012の長さ（時間）をBとする。CHUNK_0011のpresentation_start_cg_countがchunkgroup_time_base_offsetに等しく、presentation_end_cg_countが"chunk_group_time_base_offset+A"に等しい。またCHUNK_0012のpresentation_start_cg_countがchunkgroup_time_base_offset+Aに等しく、presentation_end_cg_countが"chunk_group_time_base_offset+A+B"に等しい。このように設定すると、CHUNKGROUP_002は、CHUNK_0011とCHUNK_0012を連続的に再生させたものとして定義される。

【００７０】なお、CHUNK_0011とCHUNK_0012の再生時刻に重なりがある場合には、時刻をそのようにずらすことで指定ができる。また、図２３のchunk_arrangement_info()中のtransition_info()に記述を行うことで、２つのストリーム間の遷移において、特殊効果（フェードイン、フェードアウト、ワイプ等）を指定できるようになっている。

【００７１】図２６（図２７）の例では、タイトル情報ファイル"TITLE_001.VDR"と"TITLE_002.VDR"は、チャンクグループ情報ファイル"CHUNKGROUP_001.CGIT"を、またタイトル情報ファイル"TITLE_003.VDR"はチャンクグループ情報ファイル"CHUNKGROUP_002.CGIT"を、それぞれ指定している。具体的には、図１６のtitle_info()中において、cgit_file_idというフィールドで、チャンクグループのファイルIDを指定し、さらにtitle_start_chunk_group_time_stampとtitle_end_chunk_group_time_stampというフィールドで、チャンクグループ内で該当タイトルが定義される時間的な範囲を指定している。

【００７２】例えば、図２７の例では、CHUNKGROUP_001の前半をTITLE_001が、後半をTITLE_002が、それぞれ指し示している。なお、この分割はユーザからの要求により行われたものであり、その位置はユーザにとって任意であり、予め決めておくことはできない。ここでTITLE_001とTITLE_002による分割の位置を、CHUNKGROUP_001の先頭からＡだけ離れた位置に設定したとする。

【００７３】TITLE_001はチャンクグループとしてCHUNKGROUP_001を指定し、タイトルの開始時刻として、CHUNKGROUP_001の開始時刻を指定し、タイトルの終了時刻として、ユーザから指定された点の時刻を指定する。

【００７４】つまりTITLE_001のtitle_start_chunk_group_time_stampとして、CHUNKGROUP_001のchunkgroup_time_base_offset（先頭の位置）が設定され、TITLE_001のtitle_end_chunk_group_time_stampとして、CHUNKGROUP_001のchunkgroup_time_base_offsetにＡの長さを加えたものが設定される。

【００７５】また、TITLE_002はチャンクグループとしてCHUNKGROUP_001を指定し、タイトルの開始時刻として、ユーザから指定された点の時刻を指定し、タイトルの終了時刻として、CHUNKGROUP_001の終了時刻を指定す

る。

【００７６】つまりTITLE_002のtitle_start_chunk_group_time_stampとして、CHUNKGROUP_001のchunkgroup_time_base_offset（先頭の位置）にＡの長さを加えたものが設定され、TITLE_002のtitle_end_chunk_group_time_stampとして、CHUNKGROUP_001のchunkgroup_time_base_offsetにCHUNKGROUP_001の長さを加えたものが設定される。

【００７７】さらに、TITLE_003はチャンクグループとしてCHUNKGROUP_002を指定し、タイトルの開始時刻としてCHUNKGROUP_002の開始時刻を指定し、タイトルの終了時刻としてCHUNKGROUP_002の終了時刻を指定する。

【００７８】つまりTITLE_003のtitle_start_chunk_group_time_stampとして、CHUNKGROUP_002のchunkgroup_time_base_offsetが設定され、TITLE_003のtitle_end_chunk_group_time_stampとして、CHUNKGROUP_002のchunkgroup_time_base_offsetにCHUNKGROUP_002の長さを加えたものが設定される。

【００７９】さらに、この例では、プログラム情報ファイル"PROGRAM_001.PGI"は、TITLE_001の一部とTITLE_003の一部を、この順番で再生するように指定している。具体的には、図２０のplay_item()中のtitle_numberによりタイトルを指定し、各タイトルで定義される時刻で開始点と終了点を定義することで、１つのカットが抜き出される。このようなカットを複数個集めて、プログラムが構成される。

【００８０】次に、光ディスク１に、新たな情報を追記録（アペンド記録）する場合の動作について説明する。この記録は、具体的には、例えば、タイマ録画により、あるいはユーザが入力部１４を操作して、光ディスク装置に対してリアルタイムに録画を指令することにより行われる。後者の場合、録画ボタンが押されたようなときは、録画終了時刻を予測することはできないが、ワンタッチ録画機能（操作後、一定時間だけ録画が行われる機能）のボタンが押されたときは、終了時刻を予測することができる。

【００８１】ここではタイマ録画を例にとって説明する。この場合、光ディスク装置のユーザは事前に、録画開始時刻、録画終了時刻、ビットストリームのビットレート、録画を行うチャネル等を指定してあるものとする。また、録画の予約を行った時点で、ビットレートと録画時間に見合う空き容量が光ディスク１に残されていることが、予め確認されているものとする。

【００８２】記録予約時と予約された記録の実行時との間に、光ディスク１に対して更なる記録が行われたような場合、今回予約された番組を、指定されたビットレートで記録する分の容量を確保することができなくなる場合がある。このような場合、CPU２１は、ビットレートを、指定された値より下げて、予約された時間分の情報を記録するようにするか、または、ビットレートはそのままにして、記録可能な時間だけ記録するようにする。CPU２１は、このとき、更なる記録が行われ、予約した記録に不具合が出た時点でユーザにその旨を伝えるメッセージを発することは言うまでもない。

【００８３】さて、予約された録画の開始時刻が近づくと、CPU２１は内蔵するタイマやクロックを使用して、モードを、スリープモードから動作モードに自動的に復帰させる。そしてCPU２１は、予めその処理プログラムに組み込んであるファイルシステム操作命令を使用し、予約された番組が記録できるだけの領域を光ディスク１上に確保する。つまり、予約録画の終了時刻から開始時刻を減算した結果（録画時間）にビットレートを乗じた数値が、予約された番組を記録するのに必要な領域の大きさであり、CPU２１はこの大きさの領域をまず確保する。その他、この記録に際して、ストリームファイル以外に情報ファイルを記録する必要がある場合、例えば新たなタイトルとして登録するためにタイトル情報ファイル等が必要である場合には、それらの情報ファイルが記録できるだけの容量を光ディスク１に確保しておく必要がある。必要な分の領域を確保することができない場合には、上述したような方法（ビットレートの変更、録画可能な時間内だけの録画などの方法）で対応が取られることになる。

【００８４】なおこのとき、新しいタイトルの記録なので、ユーザは、新たなストリームディレクトリの新たなストリームファイルとして新しいストリームファイルのファイル名を付ける。ここでは、これを、¥MPEGAV¥STREAMS_003¥CHUNK_0031とする。つまり、図２９に示すように、ルートディレクトリの下のMPEGAVディレクトリの下のSTREAM_003ディレクトリの下のCHUNK_0031.MPEG2という名前のファイルとする。

【００８５】CPU２１は、各部に対して記録モードの実行を命令する。例えば、図示せぬチューナから入力端子Ｐ３に入力されたビデオ信号、および入力端子Ｐ４に入力されたオーディオ信号は、エンコーダ１０によりエンコードされた後、書き込みチャネル用バッファ１１に蓄積される。続いて、CPU２１は、先程確保した領域のアドレス情報に基づき、光ヘッド２を書き込み位置に移動させる。そしてCPU２１は、光ヘッド２、ＲＦおよび復調／変調回路３、並びにECC回路４を書き込みモードに設定するとともに、スイッチ５を書き込みチャネル用バッファ１１側に切り替え、光ヘッド２の位置を微調整した後、光ヘッド２による書き込みを開始させる。これにより新たに用意した"CHUNK_0031.MPEG2"の内容が、書き込みチャネル用バッファ１１から読み出され、スイッチ５、ECC回路４、ＲＦおよび復調／変調回路３、並びに光ヘッド２を介して、光ディスク１に記録される。

【００８６】以上の書き込み動作を続けて、以下のいずれかの条件が発生した時点で、CPU２１は、書き込み動作を停止させる。

１）予約された記録の終了時刻になったとき
２）容量不足、その他の原因により光ディスク１に記録ができなくなったとき
３）録画動作の停止が指令されたとき
【００８７】次に、CPU２１は、予めその処理プログラムに組み込んであるファイルシステム操作命令を使用し、ファイルシステム中の"CHUNK_0031.MPEG2"を指し示すポインタを新しく書込んだ位置を指し示す値に書き換える。また、CPU２１は、チャンク情報、チャンクグループ情報、タイトル情報のそれぞれのファイルを用意し、しかるべき名前をつけて記録する。なお、記録時あるいは予約時に、光ディスク１上に、これらのファイルを記録することができるだけの空き容量を確保しておく必要がある。
【００８８】このようにして、例えば図３０に示すように、新たな情報ファイルが作成される。同図において、ファイル名の右肩にアスタリスク(*)をつけたものが、今回新たに作成されたファイルである。
【００８９】図３１は、新たにでき上がった情報ファイルの関係を示したものである。TITLE_004はCHUNKGROUP_003を指定し、CHUNKGROUP_003はCHUNK_0031を指定し、CHUNK_0031はSTREAM_0031を指定している。
【００９０】すなわち、新たなストリームはTITLE_004として、情報ファイルに登録されている。ユーザは光ディスク装置のタイトルを確認する機能により、TITLE_004の属性等を知ることができ、また、TITLE_004を再生することができる。
【００９１】次に、図２６（図２７）に例示するような光ディスク１上に、上書き記録する場合の動作について説明する。上書き記録とは、ビデオテープに信号を記録する場合と同様に、それまでに記録されている番組の上に（その番組を消去して）新たな番組を記録していく動作のことを言う。
【００９２】上書き記録では、上書き記録を開始する位置が重要である。例えばユーザからTITLE_001の先頭から上書き記録を開始することが指定されたとする。この時上書き記録は、TITLE_001、TITLE_002、TITLE_003をそれぞれ順に書き換えながら行われる。TITLE_003の最後まで書き換えてもまだ記録動作が終了しない場合には、光ディスク１上の空き領域に新たな領域を確保して記録が続行される。例えばTITLE_002が記録開始位置とされた場合には、TITLE_001は記録開始位置より前に位置するので、今回の記録動作により書き換えられることはない。
【００９３】いま、TITLE_003の先頭からタイマ録画により上書きするものとする。この場合、光ディスク装置のユーザは事前に、録画開始時刻、終了時刻、ビットストリームのビットレート、録画を行うチャネル等を指定しているものとする。また、上書き記録では重要な記録開始位置がTITLE_003の先頭と指定されたものとする。さらにこの場合においても、録画の予約を行った時点で、ビットレートと録画時間に見合う容量が光ディスク１上に存在することが、予め確認されているものとする。上書き記録の場合には、指定された位置から上書き可能な（複数の）タイトルの総容量と、光ディスク１の空き容量の和が記録可能容量となる。つまり、今回の場合には、TITLE_003が管理するストリームSTREAM_0011とSTREAM_0012の総容量と、光ディスク１上の空き容量の和が記録可能な容量となる。
【００９４】上書き記録では、記録可能な容量分に対して、どのような順番で実際の記録を行なっていくかという選択肢がいくつかある。まず、最初に考えられるのがタイトルで指定されているストリームの順番に記録していく方法である。つまり、今回の場合には、まずSTREAM_0011の先頭から記録を開始し、STREAM_0011の終わりまで記録したら、STREAM_0012の先頭から記録を続行し、STREAM_0012の終わりまで記録したら、今度は空き領域に記録を行なう方法である。もう１つの方法は、まず、空き領域に記録を行い、空き領域が無くなった時点で、現存するストリーム上に記録していく方法である。
【００９５】前者の方法は、ビデオテープのエミュレーションという意味で優れている。つまり、ビデオテープと同様の動作であるという意味で、ユーザから理解され易いという特徴を有する。後者の方法は、既に記録されているストリームの消去が後回しにされるため、記録されているものの保護という点で優れていると言う特徴を有する。
【００９６】なお、記録予約時と予約された記録の実行時との間に、光ディスク１に対して更なる記録が行われた場合に、今回予約された番組を、指定されたビットレートで記録する分の容量を確保することができない場合がある。このような場合、上述した場合と同様に、予約実行時に、ビットレートが自動的に下げられ、予約された時間分だけすべて記録されるか、または、ビットレートはそのままにして、記録可能な時間だけ記録が行われる。
【００９７】予約された録画の開始時刻が近づくと、光ディスク装置はスリープモードから動作モードに復帰する。CPU２１は、光ディスク１上の空き容量をすべて確保する。もちろん、この時点で空き容量を確保せず、必要になった時点で確保するという方法もあるが、ここでは説明のために、記録開始以前に必要な領域を確保するものとする。
【００９８】なお、タイマ録画等で、開始時刻、終了時刻、ビットレートが指定されているため、必要な領域の大きさが予め判っている場合には、必要な分だけ（あるいは幾分かのマージンを加えた分だけ）容量を確保するようにしてもよい。この記録に際して情報ファイルを記録する必要がある場合、例えば新たなタイトルとして登録するためにタイトル情報ファイル等が必要である場

合、それらの情報ファイルも記録することができるだけの容量を残しておく必要がある。

【００９９】ここでは、新たなストリームディレクトリの新たなストリームファイルとして新しいストリームファイルのファイル名をつけるものとする。つまり、ここでは、ファイル名を、¥MPEGAV¥STREMS_002¥CHUNK_0031とする。すなわち、図３２に示すように、ルートディレクトリの下のMPEGAVディレクトリの下のSTREAM_002ディレクトリの下のCHUNK_0031.MPEG2という名前のファイルが作成される。

【０１００】入力端子Ｐ３に入力されたビデオ信号、および入力端子Ｐ４に入力されたオーディオ信号は、エンコーダ１０によりエンコードされた後、書き込みチャネル用バッファ１１に蓄積される。続いて、CPU２１は、先程確保した領域のアドレス情報に基づき、光ヘッド２を書き込み位置に移動させる。そしてCPU２１は、光ヘッド２、ＲＦおよび復調／変調回路３、並びにECC回路４を書き込みモードに設定するとともに、スイッチ５を書き込みチャネル用バッファ１１側に切り替え、光ヘッド２の位置を微調整した後、光ヘッド２による書き込みを開始させる。これにより新たに用意した"CHUNK_0031.MPEG2"の内容が、書き込みチャネル用バッファ１１から読み出され、スイッチ５、ECC回路４、ＲＦおよび復調／変調回路３、並びに光ヘッド２を介して、光ディスク１に記録される。

【０１０１】この時、まずはストリームファイル"CHUNK_0011.MPEG2"が書き換えられる。そして"CHUNK_0011.MPEG2"の最後まで記録が行われたら、次に、"CHUNK_0012.MPEG2"へ記録が進められ、さらに、"CHUNK_0031.MPEG2"へと記録が進められる。

【０１０２】以上の動作を続けて、上述した場合と同様に、３つの条件のいずれかが発生した時点で、CPU２１は、書き込み動作を停止させる。

【０１０３】次に、CPU２１は、予めその処理プログラムに組み込んであったファイルシステム操作命令を使用し、ストリームファイル、チャンク情報、チャンクグループ情報、タイトル情報を更新する。

【０１０４】ところで、書き込みが終了したタイミングによって、ファイルの構成が変化する。例えば、CHUNK_0011.MPEG2とCHUNK_0012.MPEG2の２つのストリームの上書きを終了した後、さらにCHUNK_0031.MPEG2に記録が行われた場合、光ディスク１のファイルの構成は、図３３に示すようになる。ファイル名の右肩にアスタリスク(*)をつけたものが今回新たに作成されたファイルである。

【０１０５】図３４は、このようにして新たにでき上がったファイル（図３３のファイル）の関係を示したものである。図３１と比較して明らかなように、TITLE_003が指定しているCHUNKGROUP_002に含まれるCHUNKとしてCHUNK_0031が増えており、CHUNK_0031はSTREAM_0031を指定している。

【０１０６】一方、既存ストリームの上書きの途中で上書き記録が終了した場合、例えば、CHUNK_0011の記録の途中で上書き記録が終了した場合、上書きのために確保したCHUNK_0031のストリームは上書きされなかったので開放される。この場合、特殊なタイトルの処理が行われる。すなわち、TITLE_003の先頭から上書き記録を開始し、その途中で記録が終了した場合には、そこでタイトルが分割される。つまり、図３５に示すように、上書き記録開始位置から終了位置までが新たなTITLE_003とされ、それ以降の（元々のTITLE_003の残り部分）はTITLE_004とされる。

【０１０７】次に、タイトル再生の動作について説明する。いま、図２６に示すようなファイルを有する光ディスク１を光ディスク装置に挿入し、タイトル再生するものとする。まず、光ディスク１が挿入されると、CPU２１は情報ファイルを光ディスク１から読み込んで、RAM２４に記憶させる。この動作は上述した、基本的な情報ファイルの読み込み動作を繰り返すことで行われる。

【０１０８】CPU２１は、まず、VOLUME.TOCとALBUM.STRを読み出す。次にCPU２１は、ディレクトリ"TITLE"以下に、".VDR"の拡張子を持つファイルがいくつ有るかを調べる。この拡張子を持つファイルは、タイトルの情報を持つファイルであり、そのファイルの数はつまりタイトルの数となる。図２６の例ではタイトル数は３となる。次にCPU２１は３つのタイトル情報ファイルを読み込み、RAM２４に記憶させる。

【０１０９】CPU２１は、OSD制御回路９を制御して、光ディスク１上に記録されているタイトルの情報を示す文字情報を発生させ、合成回路８によりビデオ信号と合成させ、出力端子Ｐ１からディスプレイに出力させ、表示させる。いまの場合、タイトルが３つあること、そして３つのタイトルそれぞれの長さや属性（名前、記録された日時など）が表示される。

【０１１０】ここで、ユーザが、例えばTITLE_002の再生を指定したとする。TITLE_002の情報ファイルには（図１６のtitle_info()中のcgit_file_idには）、CHUNKGROUP_001を指定するファイルIDが記録されており、CPU２１はこれを記憶するとともに、CHUNKGROUP_001をRAM２４に格納させる。

【０１１１】次に、CPU２１は、TITLE_002の開始時刻と終了時刻（図１６のtitle_info()中のtitle_start_chunk_group_time_stampとtitle_end_chunk_group_time_stamp）が、どのCHUNKに対応するかを調べる。これは、CHUNKGROUPの情報の中から、それぞれのCHUNKが登録されている情報（図２３のchunk_arrangement_info()中のpresentation_start_cg_time_countとpresentation_end_cg_time_count）を比較することで行なわれる。いまの場合、図２７に示すように、TITLE_002の開始時刻は、CHUNK_0001の途中に入っていることがわかる。つまり、TIT

LE_002を先頭から再生するには、ストリームファイル"CHUNK_0001.MPEG2"の途中から再生を開始すれば良いと言うことがわかる。

【０１１２】次に、CPU２１は、TITLE_002の先頭がストリーム中のどこにあたるかを調べる。すなわち、TITLE_002の開始時刻が、ストリーム中のオフセット時刻（タイムスタンプ）としていくつにあたるのかが計算され、次にCHUNKファイル中の特徴点情報を使用して、開始時刻直前にあたる再生開始点が特定される。これにより、再生開始点のファイル先頭からのオフセット距離が確定できたことになる。

【０１１３】次に、CPU２１は、予めその処理プログラムに組み込んであるファイルシステム操作命令を使用し、"CHUNK_0001.MPEG2"が記録されている光ディスク１上の物理アドレスと、その長さを確定する。更に、このアドレスに、先程求めた再生開始点のオフセットアドレスが加えられて、TITLE_002の再生開始点のアドレスが最終的に確定される。

【０１１４】続いて、CPU２１は、この"CHUNK_0001.MPEG2"のアドレス情報に基づき、光ヘッド２を読み出し位置に移動させる。そしてCPU２１は、光ヘッド２、ＲＦおよび復調／変調回路３、並びにECC回路４を読み出しモードに設定するとともに、スイッチ５を読み出しチャネル用バッファ６側に切り替え、光ヘッド２の位置を微調整した後、光ヘッド２による読み出しを開始させる。これにより"CHUNK_0001.MPEG2"の内容が読み出しチャネル用バッファ６に蓄積される。

【０１１５】読み出しチャネル用バッファ６に蓄積されたデータは、デコーダ７に出力され、デコード処理が施されて、ビデオ信号とオーディオ信号が出力される。光ディスク１から読みだされ、デコードされ、表示されたデータ量が、"CHUNK_0001.MPEG2"の大きさと等しくなった時点で、CPU２１は、TITLE_003の再生に移行する。このTITLE_003の再生動作は、TITLE_002の再生動作と同様の動作である。

【０１１６】登録されているタイトルの再生が終了したとき、あるいは読み出し動作の停止が指示されたとき、読み出し、デコード処理が停止される。

【０１１７】なお、光ディスク装置に、光ディスク１として、新しいディスクが挿入された場合、あるいは、異なるフォーマットのディスクが挿入された場合、CPU２１は、ディスクが挿入されたとき、VOLUME.TOCとALBUM.STRを読み出そうとするが、これらのディスクには、このようなファイルが存在しないことになる。このような場合、即ち、VOLUME.TOCとALBUM.STRを読み出すことができない場合、CPU２１はメッセージを出力し、ユーザに指示を求める。ユーザは、CPU２１に指示し、光ディスク１をイジェクトさせるか（例えば、異なるフォーマットのディスクである場合）、初期化させるか（例えば、同一フォーマットの新しいディスクである場合）、または何らかの方法によりデータを復旧させる（例えば、同一フォーマットのディスクであるが、データが破壊されている場合）。

【０１１８】次に、titleについて、さらに説明する。図１５に示すように、TITLE_###.VDRは、titleの情報を格納するためのfileである。１つのtitleに関する情報は、１つのtitle_info()に記録される。TITLE_###.VDR内に存在するtitle_info()の数は１個である。従って、volume内にはtitleの数だけTITLE_###.VDRが存在する。

【０１１９】title numberは、図１６のtitle_info()の中で定義せず、file名またはfile idで決定される。従って、TITLE_###.VDRのうちの正の整数###がtitle numberを表す。titleは構造というよりも、chunkgroupにつけられた、開始点を表すタイトルインデックスから、次のタイトルの先頭を表すタイトルインデックスまでの範囲、またはchunkgroupの終了点までの範囲の部分である。

【０１２０】図１５のTITLE_###.VDRのfile_type_idは、図３６に示すように、title_info()が記録されたfileであることを示すidであり、長さ１６の文字列で表される。text_block()は、さまざまなtextを格納するための領域であり、そのtext_block()で使用が許されているtext itemだけが記述される。

【０１２１】title_info()は、図１６に示すように、chunkgroup上における、titleの開始点と終了点、その他のtitleに関する属性が書かれる領域である。また、title_info()は、タイトル番号順に再生したとき、タイトル間でシームレス再生が保証できるか否かを示すflagを持つことができる。このflagにより、光ディスク装置でタイトル間をシームレス再生できるか否かが事前に把握でき、また併合時に配置を変える必要があるかどうかがわかる。

【０１２２】タイトルの境界はfileの境界でも有り得るため、シームレス再生は保証されないことがある。ただし、光ディスク装置の機能として、再配置等を行うことにより、一般的にシームレス再生が行われるような状態にすることは可能である。

【０１２３】図１６のtitle_info()中のtitle_info_lengthは、title_info()の長さをbyte単位で表したものである。flags_for_titleには、対応するtitleの書き込み属性、書き込み（変更許可）、再生回数制限、ratingのlevel等が記録される。cgit_file_idには、対応するtitleのbaseであるchunkgroupのinformation file（CHUNKGROUP_###.CGIT）のfile_idが記録される。

【０１２４】title_start_chunk_group_time_stampには、chunkgroupで定義されたlocalな時間軸上における、そのtitleの再生開始点の時刻が記録される。そのtitleのtitleindexが指しているピクチャの表示時間がこの値となる。title_end_chunk_group_stampには、chunkgroupで定義されたlocalな時間軸上における、そのtitl

eの再生終了点の時刻が記録される。この値は、chunkgroupの再生終了点か、または時間軸上において直後に位置するtitleの開始点を表すtitleindexが示す値と同一となる。

【０１２５】title_playback_time()には、そのtitleの再生時間（タイムコード値、またはframe若しくはfield枚数）が記録される。number_of_marksには、そのtitle内に設定されているすべてのmarkの総数（titleindexを除く）が記録される。mark_typeには、図３７に示すように、title内の任意の位置につけられるmarkの種類が記録される。markは、title内のrandom access pointとしても利用される。mark_chunk_group_time_stampには、chunkgroupの時間軸上において、そのmarkが設定されている個所のtime stampが、値の小さなものから順に記録される。time stampが同じ値のmarkが複数存在してはならない。titleの開始点および終了点と同じtime stampを有するindexは存在してもよい。stuffing_bytesには、stuffingするbytesが記録され、その長さは８×n bit(n>;=0)となる。

【０１２６】次に、図２１乃至図２４に示したchunkgroupとchunkについてさらに説明する。CHUNKGROUP_###.CGITは、titleの時間軸の定義、およびchunkの構成、titleに含まれる不連続点の処理を記述したファイルである。

【０１２７】titleは各種のbitstreamで構成されており、videoが無いstreamである場合や、DV（ディジタルビデオ）のbitstreamである場合もある。DVフォーマットでは、フレーム単位で時間軸が規定されており、MPEG2 videoのSTC（System Time Clock）を基準にしていると、フォーマットが異なるので、このDVのbitstreamを管理することができない。

【０１２８】そこで、title内でlocalな時間軸を設定するものとする。この時間軸はtitleを構成するstreamに依存しない。titleの境界はchunkの境界とは無関係に設定される。そのため、localな時間軸は、chunk（bitstreamに１対１で対応される）毎や、title毎に設定するよりも、複数の（任意の数の）titleが含まれるchunkの集合体に対して設定するのが適切である。このchunkの集合体がchunkgroupである。

【０１２９】chunkgroupでは、単一の時間軸を定義し、その上にchunkを貼り付けていくことでchunkの表示時刻を定めている。つまり、chunkgroupには、bitstream fileの内容（byte列）を時間軸上に展開した状態でchunkが並ぶ。１つのbitstream fileに含まれるすべてのchunkを時間軸上に並べたものをpathと呼ぶ。chunkgroupには、複数のpathを並べることができる。pathのうち、chunkgroupの再生開始時刻と終了時刻を規定するpathはmain pathと称され、その他のpathは、sub pathと称される。sub pathは、主に、後から追加記録されたaudioのchunk等を表す。

【０１３０】chunkの接続点はtitleの境界とは必ずしも一致しないので、titleの属性ではない。しかし、chunk間の関係を各chunkの属性の含めると、階層的に矛盾が発生する場合がある。このような不連続点情報は、chunkとtitleの中間に位置するものであり、chunkgroupの階層に置くのが適切であると考えられる。

【０１３１】以上をまとめると、chunkgroupが持つ情報は、chunkの時間軸上への配置の仕方、chunkの再生順序、chunkの終わりと次に再生するchunkの始まりとの接続点で発生する不連続点などである。

【０１３２】図２１のCHUNKGROUP_###.CGITのfile_type_idは、図３８に示すように、そのファイルがCHUNKGROUP_###.CGITであることを表す識別子であり、ISO 646に従った１６文字の文字列で表される。chunkgroup_time_base_flagsには、chunkgroupの基準カウンタに関するflagが記録される。chunkgroup_time_base_offsetには、chunkgroup内の基準時間軸の開始時刻が記録される。この値は、９０kHzのクロックをカウントアップするカウンタにセットされる値であり、３２bitで表される。text_block()は、さまざまなtextを格納するための領域であり、そのtext_block()で使用が許されているtext itemだけが記述される。

【０１３３】図２２に示すように、chunk_connection_info()は、特異な点の情報（videoの切り替え点、videoとaudioの同期など）を記録しておくためのファイルであり、chunk間の接続状況を規定している。編集によりできたchunkとchunkのつなぎ目のような特異点では、GOPの途中でchunkを乗りかえる必要がある。このような編集点付近の情報がここに記述される。chunkは２つ以上のchunkgroupに属することはない。

【０１３４】chunk_connection_info_lengthには、chunk_connection_info()の長さをbyte単位で表したものが記録される。number_of_chunksには、そのchunkgroupで使われるchunkの総数が記録される。chunk_sync_play_flagは、図３９に示すように、同時刻に２つ以上のchunkを再生する必要があるかどうかを表すflagであり、その値の０は、１つのchunkの再生を意味し、その値の１は、複数のchunkの同時再生を意味する。

【０１３５】図２３のchunk_arrangement_info()において、chunk_arrangement_info_lengthには、各chunkについての情報の長さをbyte単位で表したもの（chunk_arrangement_info_lengthの先頭byteからtransition_info()の最終byteまでを含めた長さ）が記録される。chunk_info_file_idには、対象となるchunk_info_fileのfile_idが記録される。

【０１３６】chunk_switch_stream_idには、接続したとき連続再生するstreamのstream_idが記録される。このidとしては、例えば、MPEG2のパケットヘッダに記録されているビデオあるいはオーディオを識別するidが用いられる。presentation_start_cg_time_countには、該当ch

unkの表示開始時刻を、chunkgroup内の時刻で表したtime count値が記録される。chunkの表示開始時刻は、chunkgroup内で定義されるglobalなtime stampで表現される。該当chunkは、chunkgroupにおいて、この時刻から表示が開始される。presentation_end_cg_countには、該当chunkの表示終了時刻を、chunkgroup内の時刻で表したtime count値が記録される。chunkの表示終了時刻は、chunkgroup内で定義されるglobalなtime stampで表現される。

【０１３７】図４０に示すように、original_time_count_typeには、stream内部で使われているtime countの種類が記録される。例えば、MPEG2 videoのstreamであれば、original_time_count_typeは’００００’とされる。number_of_start_original_time_count_extensionには、複数のtime countが必要なとき、新たに必要な開始時刻を表すtime countの数が記録される。number_of_end_original_time_count_extensionには、複数のtime countが必要なとき、新たに必要な終了時刻を表すtime countの数が記録される。presentation_start_original_time_countには、presentation_start_tg_time_countに対応する、stream内部における時刻あるいはカウンタ値が記録される。presentation_end_original_time_countには、presentation_end_tg_time_countに対応する、stream内部における時刻あるいはカウンタ値が記録される。

【０１３８】tc_ext_attributesには、time_count_extensionに対する属性が記録される。このtime_count_extensionには、例えば、どのstreamに適用するのか等の情報を入れることができる。start_original_time_count_extensionには、chunkの切り替えに必要な開始時刻あるいは開始カウンタ値が記録される。これはオプションであり、複数の時刻やカウンタ値を記録する必要があるときに使用される。end_original_time_count_extensionには、chunkの切り替えに必要な終了時刻あるいは終了カウンタ値が記録される。これもオプションであり、複数の時刻やカウンタ値を記録する必要があるときに使用される。transition_info()には、chunkの切り替えで特殊効果をかけるときに必要な情報が記録される。例えば、chunkの指定、切り替え時刻、特殊効果の種類等がここに記述される。

【０１３９】図２４に示すように、CHUNK_%%%%.ABSTは、sub_file番号%%%%のchunkを構成するbitstreamから抽出した特徴点を記録したfileである。このfileには、GOP,Audio frame等のbitstreamを構成する単位毎に、その開始byte位置、長さ、属性等が記述される。GOP情報、Audio frame情報はchunk(sub-file)毎に、１つのCHUNK_%%%%.ABSTとしてまとめられる。

【０１４０】図４１に示すように、CHUNK_%%%%.ABSTのfile_type_idには、stream_info()が記録されているfileであることを示す識別子が、ISO 646に従った１６文字の文字列で記録される。

【０１４１】図４２に示すように、info_typeには、図２４において次に続くstream_infoのtypeが記録される。ここでstreamの種類が特定される。number_of_programsには、MPEG2のTS(Transport Stream)に含まれるprogramの数が記録される。この数を知るには、PSI(Program Specific Information)を読み取る必要がある。TS以外のときには、この値は１になる。number_of_streamsには、そのprogramで使われるストリームの数が記録される。この値は、TSの場合には、異なるPID(packet identification)の数と等しくなる。TS以外のMPEGstreamの場合、ここには、stream idが異なるstreamの数が記録される。

【０１４２】stream_identifierには、stream idが記録される。TSの場合には、stream idとして、PIDが利用される。

【０１４３】図４３に示すように、slot_unit_typeには、streamをある一定間隔毎に区切ったときの、その区切り方が記録される。各frame, field等、区切りの指標が時間の場合には、time stamp valueが用いられる。slot_time_lengthには、１slotに対応する時間が記録される。この値は、９０kHzのクロックをカウントするカウンタを用いたtime stampの値で表される。number_of_slotsには、CHUNK_%%%%.ABSTに書かれているslot_info()の数が記録される。number_of_I_pictures_in_a_slotには、slotに含まれるI-pictureの数が記録される。この値は、１以上の整数で１５以下の値となる。ただし、GOPheaderを先頭とするslotの直前に位置するslotに含まれるI-pictureの数は、この値よりも小さくてもよい。GOPheaderの直後でないI-pictureのpicture headerを先頭とするslotを設定するとき、この値が活用される。

【０１４４】次に、図１７と図１８に示したprogramについてさらに説明する。PROGRAM_$$$.PGIには、program()がただ１つ存在する。volume内にはprogramの数だけPROGRAM_$$$.PGIが存在する。program番号は、program()の中で定義せず、file名またはfile idで規定される。

【０１４５】図４４に示すように、図１７のPROGRAM_$$$.PGIのfile_type_idには、program()が記録されたfileであることを示すidが、長さ１６の文字列で記録される。text_block()には、さまざまなtextを格納するための領域が形成されている。ここには、そのtext_block()で使用が許されているtext itemだけが記述される。

【０１４６】図１８のprogram()のflags_for_programには、programに関する各種フラグが記録される。例えば、このprogramの書き込み属性、書き込み（変更許可）、再生回数制限、ratingのlevel等が記録される。

【０１４７】図４５に示すように、program_statusには、programの属性が記録される。このfieldの設定はoptionであるが、設定しないときは"none"にしなければならない。

【０１４８】program_playback_time()には、そのprogramの再生時間が記録される。number_of_play_sequencesには、そのprogramで使用されているplay_sequenceの数が記録される。ただし、このformatの例では、値は１に固定されている。すなわち、このformatの例では、１program＝１ch（チャンネル）再生とされているので、２ch同時再生を実現するには、２programの同時再生指定を可能にすればよい。１program＝１ch再生の制限がなければ、１programで、２ch同時再生も可能である。multichannel Ｉ／Ｏを使って２つのplay sequenceを同時再生するとき、どのplay sequenceをどのoutput channelに割り当てるかは、光ディスク装置が決める。

【０１４９】number_of_play_listsには、このplay sequenceで使用されているplay_listの数が記録される。この例では値は１とされる。play_list_start_time_stamp_offsetには、play sequenceの開始時刻から開始するtimerでカウントした、play sequence内での時刻が記録される。この値が、play listの開始時刻になる。programでは、play sequence内にplay listは１つしか存在してはならない。時刻の単位系は９０kHzである（１／９００００秒が時刻の最小単位である）。stuffing_bytesには、stuffingのbytesが記録される。その長さは、８×n bit(n>=0）とされる。

【０１５０】次に、不連続点フラグについてさらに説明する。ここで不連続点フラグとは、図１９のplay_list()のseamless_connection_flag、または図３７に示したindextype 8として記録されるマークを意味する。

【０１５１】seamless_connection_flagの値の０は、図４６に示すように、直前のplay itemとの連続再生（シームレス再生）が保証されていないこと、または不明であることを意味し、その値の１は、シームレス再生が保証されていることを意味する。

【０１５２】すなわち、図４７に示すように、このflagが０である場合には、所定のplay itemが直前のplay itemから連続して（画像あるいは音声を途切れさせることなく）再生することが保証されている。これに対して、このflagが１である場合には、直前のplay itemが終了した後、次のplay itemが再生されるまでの間に不連続部分が発生する場合があることになる。

【０１５３】次に、図４８のフローチャートを参照して、titleの不連続点フラグを記録する処理を説明する。最初にステップＳ１において、ユーザは、入力部１４を操作して、記録対象とするtitleを指定する。このとき、ステップＳ２において、CPU２１は、光ディスク１にステップＳ１で指定されたtitleを記録したとき、そのtitleを、既に記録されている直前のtitleに対して連続して再生することが可能であるか否かを判定する。この判定は、読み出しチャネル用バッファ６の容量（その容量のデータを、デコーダ７がデコードするのに要する時間）と、title間の再生時間に対応して行われる。すなわち、次のtitleを再生するまでの間に、読み出しチャネル用バッファ６が空になってしまうときは、連続再生が不可能と判定され、空にならないときは、連続再生が可能と判定される。

【０１５４】連続再生が可能でない場合には、ステップＳ３に進み、CPU２１は、OSD制御回路９を制御し、いま記録が指令されたtitleは、既に記録されているtitleに対して、連続再生することができないこと、すなわち、両者の間には、不連続点が発生することを表すメッセージを発生させる。このメッセージは、合成回路８から、出力端子Ｐ１を介して、ディスプレイに表示される。

【０１５５】ユーザは、このメッセージを見て、不連続点の発生を了承するか否かを入力部１４を操作して入力する。ユーザが不連続点の発生を了承した場合、CPU２１は、ステップＳ５において、図１６のtitle_info()中のmark_typeに、図３７に示すマークのうちの、不連続点を示すインデックスとしてのindex type 8を設定させるとともに、不連続点を発生させる位置を、図１６のrelative_time_stamp_in_titleに設定する処理を実行する。なお、このrelative_time_stamp_in_titleに記録する位置（不連続点を発生させる位置）は、任意の位置とすることができる。従って、通常、不連続点が発生しても影響が少ない点（例えば、次のtitleの先頭、または、前のtitleの最後）がここに記録される。

【０１５６】そして、このような設定が行われたtitle_info()は、RAM２４から書き込みチャネル用バッファ１１に供給され、記憶された後、所定のタイミングで、そこから読み出され、スイッチ５、ECC回路４、ＲＦおよび復調／変調回路３、および光ヘッド２を介して、光ディスク１に供給され、記録される。

【０１５７】次に、ステップＳ６に進み、CPU２１は、title内に他にも連続して再生することができない点が存在するか否かを判定し、再生不連続点が他にも存在する場合には、ステップＳ３に戻り、それ以降の処理を繰り返し実行する。title内に再生不連続点が他には存在しないと判定された場合、処理は終了される。

【０１５８】一方、ステップＳ４において、ユーザが不連続点の発生を了承しない旨を入力した場合、ステップＳ７に進み、CPU２１は、既に記録済みのtitleの記録位置を、これから記録するtitleと連続再生可能な位置に変更可能か否かを判定する。titleの記録位置を変更可能であると判定した場合、CPU２１は、ステップＳ８に進み、既に記録されているtitleの記録位置を、これから記録するtitleと連続再生可能な位置に変更する。その後、ステップＳ６に進む。

【０１５９】ステップＳ７において、titleの記録位置を変更することができないと判定された場合、ステップＳ５に進み、上述したように、mark_typeとして不連続点が記録されるとともに、不連続点の発生位置がrelative_time_stamp_in_titleに記録される。

【０１６０】このような処理の後、ステップＳ１で指定されたtitleの記録処理が実行される。

【０１６１】次に、図４９のフローチャートを参照して、program作成時に不連続点フラグを記録する場合の処理について説明する。最初にステップＳ２１において、ユーザは、入力部１４を操作して、programに含めるtitleを指定する。この指定が行われたとき、ステップＳ２２において、CPU２１は、ステップＳ２１で指定されたtitleの中の必要な部分を、再生開始点および再生終了点で指定する。CPU２１は、この指定に対応して、play item（図２０）を作成する。

【０１６２】次に、ステップＳ２３において、CPU２１は、直前のplay itemとの間で連続再生が可能か否かを判定し（この判定も、play itemとplay itemの間に発生する再生時間と、読み出しチャネル用バッファ６の容量（その容量のデコード時間）との比較に基づいて行われる）、連続再生が可能でない場合には、ステップＳ２４に進み、OSD制御回路９を制御し、不連続点が発生することをメッセージで表示させる。このメッセージに対して、ユーザは、入力部１４を操作して、不連続点の発生を了承するか否かを入力する。そこで、CPU２１は、ステップＳ２５において、ユーザが不連続点の発生を了承したか否かを判定し、ユーザが不連続点の発生を了承したと判定した場合には、ステップＳ２６に進み、そのplay itemのseamless_connection_flagを０に設定する。図４６を参照して説明したように、このflagが０であるということは、連続再生が保証されていないことを意味する。

【０１６３】次に、ステップＳ２７に進み、CPU２１は、programの作成処理が終了したか否かを判定し、まだ終了していないと判定した場合には、ステップＳ２３に戻り、それ以降の処理を繰り返し実行する。ステップＳ２７において、programの作成処理を終了したと判定した場合、処理は終了される。

【０１６４】一方、ステップＳ２５において、ユーザが、不連続点の発生を了承しないと判定された場合、ステップＳ２８に進み、CPU２１は、対象としているストリームの光ディスク１上の記録位置（配置位置）を変更するか、あるいは部分的再記録が可能であるか否かを判定する。ストリームの再配置処理または部分的再記録が可能である場合には、ステップＳ２９に進み、CPU２１は、記録位置を連続再生可能な位置に変更する処理を実行する。そして、この場合には、ステップＳ３０において、CPU２１は、そのplay itemのseamless_connection_flagを１に設定する。図４６を参照して説明したように、このflagの１は、連続再生が保証されていることを意味する。その後、ステップＳ２７に進み、それ以降の処理が実行される。

【０１６５】ステップＳ２８において、ストリームの再配置処理または部分的再記録が不可能であると判定された場合には、ステップＳ２６に進み、ユーザが不連続点の発生を了承した場合と同様の処理が実行される。

【０１６６】ステップＳ２３において、直前のplay itemとの間で連続再生が可能であると判定された場合には、ステップＳ３０に進み、直ちに、そのplay itemのseamless_connection_flagに１が設定される。

【０１６７】次に、図４８を参照して説明したように、titleに不連続点フラグが記録された場合に、その再生が指令されたときの処理について、図５０のフローチャートを参照して説明する。最初に、ステップＳ４１において、CPU２１は、指定されたtitleの不連続点フラグを読み取る。ステップＳ４２において、CPU２１は、titleの再生処理を開始し、ステップＳ４３において、再生を開始したtitleが終了したか否かを判定する。titleの再生が終了していない場合には、ステップＳ４４に進み、CPU２１は、不連続点フラグ（relative_time_stamp_in_title）で表された位置が再生されたか否かを判定し、まだ再生されていないと判定された場合には、ステップＳ４３に戻る。

【０１６８】ステップＳ４４において、不連続点フラグで表された位置が再生されたと判定された場合、ステップＳ４５に進み、CPU２１は、プレーヤ（いまの場合、光ディスク装置）が後続部分の連続再生のため、ギャップが必要であるか否かを判定する。この判定は、読み出しチャネル用バッファ６に、その時点において記憶されているデータ量（そのデータ量をデコーダ７がデコードするのに要する時間）に基づいて行われる。そのデータ量が充分である場合には、ギャップは不要と判定され、そのデータ量が不充分である場合には、ギャップが必要と判定される。この光ディスク装置が後続部分の連続再生のためギャップを必要としていない場合（読み出しチャネル用バッファ６にデータが充分記憶されている場合）には、ステップＳ４３に戻り、それ以降の処理が繰り返し実行される。

【０１６９】これに対して、ステップＳ４５において、光ディスク装置が後続部分の再生のためにギャップを必要としていると判定された場合（読み出しチャネル用バッファ６にデータが充分記憶されていない場合）、ステップＳ４６に進み、CPU２１は、ギャップを発生させる。すなわち、CPU２１は、読み出しチャネル用バッファ６に対するデータの書き込みはそのまま継続させるが、デコーダ７に対するその読み出しは中断させ、読み出しチャネル用バッファ６のデータ量を増加させる。そして、ステップＳ４７に進み、後続部分の連続再生が可能になったか否かを判定し、まだ可能になっていない場合には、ステップＳ４６に戻り、ギャップ発生処理を継続させる。

【０１７０】以上のようにして、所定量のデータが読み出しチャネル用バッファ６に書き込まれ、ステップＳ４７において、後続部分の連続再生が可能な状態になった

と判定された場合、ステップＳ４８に進み、CPU２１は、再び再生を継続（デコーダ７によるデコードを再開）させる。その後、ステップＳ４３に戻り、それ以降の処理が繰り返し実行される。

【０１７１】一方、ステップＳ４３において、titleが終了したと判定された場合、ステップＳ４９に進み、次に再生するtitleが存在するか否かをCPU２１は判定する。次に再生するtitleが存在しない場合には、処理は終了される。これに対して、次に再生するtitleが存在すると判定された場合には、ステップＳ５０に進み、CPU２１は、前のtitleの最後に不連続フラグが存在するか否かを判定する。前に再生したtitleの最後に不連続フラグが存在すると判定された場合、ステップＳ５１に進み、CPU２１は、プレーヤ（光ディスク装置）は次のtitleの再生のためにギャップを必要とするか否かを判定する。ギャップが必要であると判定された場合、ステップＳ５２に進み、CPU２１は、ギャップ発生処理を実行し、ステップＳ５３において、次のtitleの再生が可能になったか否かを判定し、可能になっていない場合には、ステップＳ５２に戻り、ギャップ発生処理を繰り返し実行する。

【０１７２】ステップＳ５３において、次のtitleの再生が可能になったと判定された場合、ステップＳ４１に戻り、それ以降の処理が繰り返し実行される。

【０１７３】ステップＳ５０において、前のtitleの最後に不連続フラグが存在しないと判定された場合、ステップＳ４１に戻り、それ以降の処理が実行される。また、前のtitleの最後に不連続フラグが存在したとしても、ステップＳ５１において、光ディスク装置が次のtitleの再生のためにギャップを必要としないと判定された場合（その光ディスク装置の読み出しチャネル用バッファ６の容量が充分大きい場合）には、ステップＳ４１に戻り、それ以降の処理が実行される。

【０１７４】次に、図４９に示すようにして、programが作成された場合に、そのprogramの再生が指令されたときの処理について、図５１のフローチャートを参照して説明する。

【０１７５】最初に、ステップＳ６１において、CPU２１は、指定されたprogramを構成する最初のplay itemを再生する。ステップＳ６２において、CPU２１は、play itemの再生が終了したか否かを判定し、まだ終了していない場合には、終了するまで待機する。

【０１７６】ステップＳ６２において、play itemの再生が終了したと判定された場合、CPU２１は、ステップＳ６３に進み、次のplay itemが存在するか否かを判定する。次のplay itemが存在すると判定された場合、ステップＳ６４に進み、CPU２１は、次のplay itemのseamless_connection_flagが０であるか否かを判定する。このflagが０であると判定された場合（シームレス再生が保証されていない場合）、ステップＳ６５に進み、CPU２１は、光ディスク装置が次のplay item再生のためにギャップを必要とするか否かを判定し、ギャップを必要とする場合には、ステップＳ６６に進み、ギャップ発生処理を実行する。そして、ステップＳ６７において、次のplay itemを再生することが可能な状態になったか否かを判定し、まだ可能な状態になっていない場合には、ステップＳ６６に戻り、ギャップ発生処理を繰り返し実行する。

【０１７７】ステップＳ６７において、次のplay itemを再生することが可能な状態になったと判定された場合（読み出しチャネル用バッファ６に充分な量のデータが書き込まれた場合）、ステップＳ６８に進み、CPU２１は、次のplay itemの再生処理を開始する。その後、ステップＳ６２に戻り、それ以降の処理が繰り返し実行される。

【０１７８】ステップＳ６４において、次のplay itemのseamless_connection_flagが０ではないと判定された場合（１であると判定された場合）、あるいはまた、ステップＳ６５において、光ディスク装置が次のplay item再生のためにギャップを必要としていないと判定された場合、ステップＳ６６とステップＳ６７の処理はスキップされ、ステップＳ６８に進む。そして、次のplay itemの再生が開始され、その後、ステップＳ６２に戻る。

【０１７９】以上においては、本発明を光ディスク装置に応用した場合を例として説明したが、本発明は、その他の記録媒体に情報を記録または再生する場合にも適用することが可能である。

【０１８０】なお、上記したような処理を行うコンピュータプログラムをユーザに提供する提供媒体としては、磁気ディスク、CD-ROM、固体メモリなどの記録媒体の他、ネットワーク、衛星などの通信媒体を利用することができる。

【０１８１】

【発明の効果】以上の如く、請求項１に記載の記録再生装置、請求項４に記載の記録再生方法、および請求項５に記載の提供媒体によれば、データを既に記録されているデータに対して連続して再生可能であるか否かを判定し、その判定結果に対応して、連続再生に関する処理を実行するようにしたので、互換性を確保可能な記録媒体を実現することができる。

【０１８２】請求項６に記載の記録再生装置、請求項７に記載の記録再生方法、および請求項８に記載の提供媒体によれば、記録媒体から抽出された連続再生情報に対応して、ギャップを付加するようにしたので、記録再生装置のバッファの容量の違いに拘らず、互換性を確保することができる。従って、ユーザが装置の故障と誤認するのを抑制することができる。

【図面の簡単な説明】

【図１】ディレクトリの構造を説明する図である。

【図２】VOLUME.TOCを説明する図である。
【図３】volume_information()を説明する図である。
【図４】volume_attribute()を説明する図である。
【図５】resume()を説明する図である。
【図６】volume_rating()を説明する図である。
【図７】write_protect()を説明する図である。
【図８】play_protect()を説明する図である。
【図９】recording_timer()を説明する図である。
【図１０】text_block()を説明する図である。
【図１１】language_set()を説明する図である。
【図１２】text_item()を説明する図である。
【図１３】ALBUM.STRを説明する図である。
【図１４】album()を説明する図である。
【図１５】TITLE_###.VDRを説明する図である。
【図１６】title_info()を説明する図である。
【図１７】PROGRAM_$$$.PGIを説明する図である。
【図１８】program()を説明する図である。
【図１９】play_list()を説明する図である。
【図２０】play_item()を説明する図である。
【図２１】CHUNKGROUP_###.CGITを説明する図である。
【図２２】chunk_connection_info()を説明する図である。
【図２３】chunk_arrangement_info()を説明する図である。
【図２４】CHUNK_%%%%.ABSTを説明する図である。
【図２５】本発明を適用した光ディスク装置の構成例を示すブロック図である。
【図２６】ディレクトリの構造を説明する図である。
【図２７】ディレクトリの論理構造を説明する図である。
【図２８】offsetを説明する図である。
【図２９】ディレクトリの構造を説明する図である。
【図３０】ディレクトリの構造を説明する図である。
【図３１】ディレクトリの論理構造を説明する図である。
【図３２】ディレクトリの構造を説明する図である。
【図３３】ディレクトリの構造を説明する図である。
【図３４】ディレクトリの論理構造を説明する図である。
【図３５】ディレクトリの論理構造を説明する図である。
【図３６】file_type_idを説明する図である。
【図３７】mark_typeを説明する図である。
【図３８】file_type_idを説明する図である。
【図３９】chunk_sync_play_flagを説明する図である。
【図４０】original_time_count_typeを説明する図である。
【図４１】file_type_idを説明する図である。
【図４２】info_typeを説明する図である。
【図４３】slot_unit_typeを説明する図である。
【図４４】file_type_idを説明する図である。
【図４５】program_statusを説明する図である。
【図４６】seamless_connection_flagを説明する図である。
【図４７】seamless_connection_flagの意味を説明する図である。
【図４８】titleの不連続点フラグ記録処理を説明するフローチャートである。
【図４９】program作成時の不連続点フラグの記録処理を説明するフローチャートである。
【図５０】title再生時の不連続点フラグの処理を説明するフローチャートである。
【図５１】program再生時の不連続点フラグの処理を説明するフローチャートである。

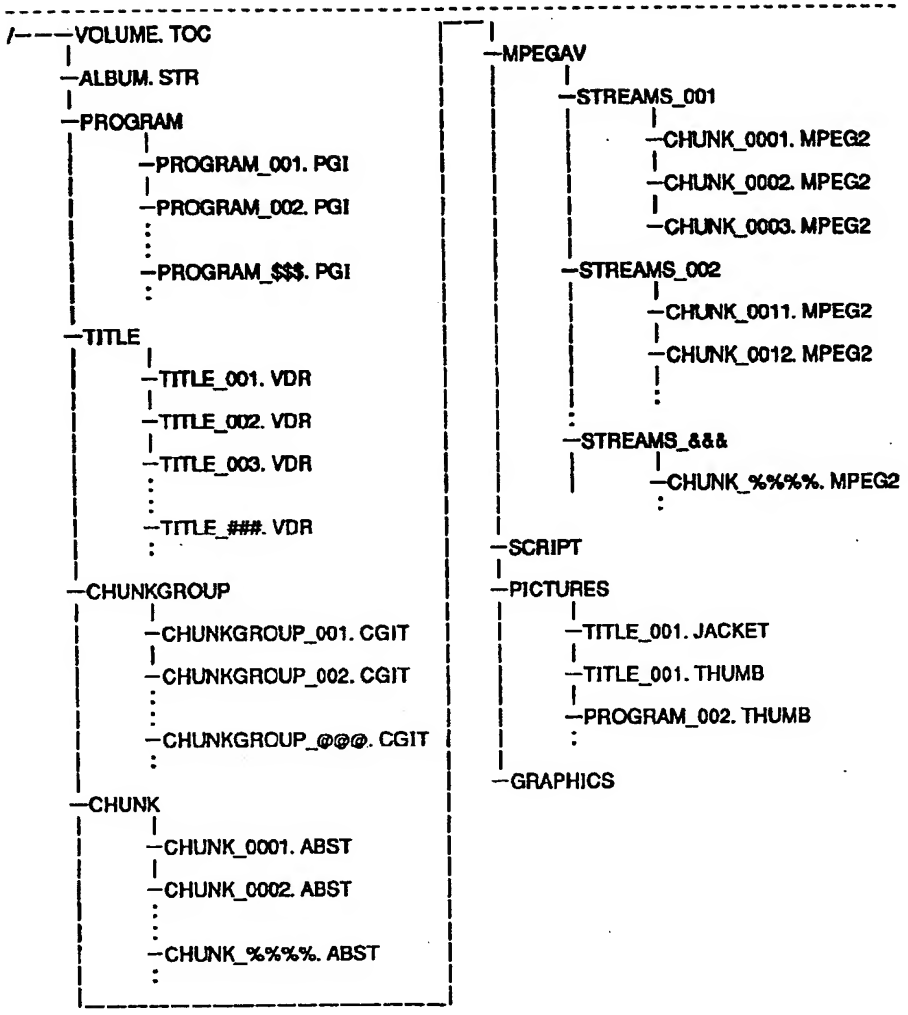
【符号の説明】

１　光ディスク，　２　光ヘッド，　３　ＲＦおよび復調／変調回路，　４ECC回路，　６　読み出しチャネル用バッファ，　７　デコーダ，　８　合成回路，　９　OSD制御回路，　１０　エンコーダ，　１１　書き込みチャネル用バッファ，　１２　アドレス検出回路，　１３　システムコントローラ，　１４入力部，　２１　CPU，　２２　ROM，　２３，２４　ＲＡＭ

【図２】

Syntax	Number of Bits	Mnemonic
VOLUME.TOC {		
file_type_id	8*16	char[16]
volume_information ()		
text_block ()		
}		

【図1】



【図3】

Syntax	Number of Bits	Mnemonic
volume_information () {		
volume_attribute ()		
resume ()		
volume_rating ()		
write_protect ()		
play_protect ()		
recording_timer ()		
}		

【図4】

Syntax	Number of Bits	Mnemonic
volume_attribute () {		
volume_attribute_length	32	uimsbf
vdr_version	4*4	bcd
reserved	6	bslbf
title_playback_mode_flag	1	bslbf
program_playback_mode_flag	1	bslbf
volume_play_time ()	4*8	bcd
update_time_count ()	32	uimsbf
maker_id	8*16	char[16]
model_code	8*16	char[16]
POSID	32	bslbf
}		

【図6】

Syntax	Number of Bits	Mnemonic
volume_rating () {		
volume_rating_length	32	uimsbf
reserved	6	bslbf
volume_rating_type	2	bslbf
volume_rating_password	128	bslbf
switch (rating_type) {		
case age_limited :		
country_code_for_rating	32	bslbf
for (i=0; i<32; i++) {		
rating_bit_for_age_limited	1	bslbf
break ;		
case CARA :		
CARA_category	4	bslbf
reserved	4	bslbf
reserved	16	bslbf
break ;		
case RSAC :		
RSAC_category	4	bslbf
level	4	bslbf
reserved	16	bslbf
break ;		
}		
}		

【図13】

Syntax	Number of Bits	Mnemonic
ALBUM. STR {		
file_type_id	8*16	char[16]
album ()		
text_block ()		
}		

【図５】

Syntax	Number of Bits	Mnemonic
resume () {		
resume_length	32	uimsbf
reserved // for byte alignment	3	bslbf
resume_switch	1	bit
reserved	4	bslbf
number_of_records	4	uimsbf
reserved // for byte alignment	7	bslbf
resume_auto_execute_time_flag	1	bit
resume_auto_execute_time ()	4*14	bcd
reserved	4	bslbf
resume_auto_execute_record_number	4	uimsbf
for (i=0; i<number_of_records; i++) {		
resume_mode_flag	4	bslbf
object_type	4	bslbf
linked_record_number	4	uimsbf
number_of_times	16	uimsbf
resume_updated_time ()	4*14	bcd
switch (object_type) {		
case title :		
title_number	16	uimsbf
title_local_time_stamp	64	uimsbf
break ;		
case program :		
program_number	16	uimsbf
program_local_time_stamp	64	uimsbf
break ;		
case program_bind :		
program_bind_number	16	uimsbf
program_order	16	uimsbf
program_number	16	uimsbf
program_local_time_stamp	64	uimsbf
break ;		
case play_item :		
play_item_number	16	uimsbf
play_item_local_time_stamp	64	uimsbf
break		
}		
}		
}		

【図１５】

Syntax	Number of Bits	Mnemonic
TITLE_###. VDR {		
file_type_id	8*16	char[16]
title_info ()		
text_block ()		
}		

【図7】

Syntax	Number of Bits	Mnemonic
write_protect () {		
write_protect_length	32	uimsbf
volume_write_protect_level	4	uimsbf
password_enable_flag	1	bslbf
append_only_flag	1	bslbf
expiration_time_enable_flag	1	bslbf
number_of_times_enable_flag	1	bslbf
password_for_volume_write_protect	128	bslbf
reserved	8	bslbf
write_protect_set_time ()	56	bcd
reserved	8	bslbf
write_protect_expiration_time ()	56	bcd
number_of_times	16	uimsbf
}		

【図8】

Syntax	Number of Bits	Mnemonic
play_protect () {		
play_protect_length	32	uimsbf
volume_play_protect_flag	2	bslbf
reserved	2	bslbf
password_enable_flag	1	bslbf
reserved	1	bslbf
expiration_time_enable_flag	1	bslbf
number_of_times_enable_flag	1	bslbf
password_for_volume_play_protect	128	bslbf
reserved	8	bslbf
play_protect_set_time ()	56	bcd
reserved	8	bslbf
play_protect_expiration_time ()	56	bcd
number_of_times	16	uimsbf
}		

【図9】

Syntax	Number of Bits	Mnemonic
recording_timer () {		
recording_timer_length		
recording_timer_flag		
number_of_entry		
for (i=0; i<number_of_entry; i++) {		
date_and_time		
channel		
program		
:		
}		
}		

【図10】

```
Syntax                                                  Number of Bits    Mnemonic
text_block () {
        text_block_length                                   32            uimsbf
        number_of_language_sets                             8             uimsbf
        number_of_text_items                                16            uimsbf
        for (i=0; i<number_of_language_sets; i++) {
            language_set ()
        }
        for (i=0; i<number_of_text_items; i++) {
            text_item ()
        }
}
```

【図11】

```
Syntax                                                  Number of Bits    Mnemonic
language_set () {
        reserved                                            8             bslbf
        language_code                                       24            bslbf
        number_of_language_set_names                        8             uimsbf
        for (i=0; i<number_of_language_set_names; i++) {
            character_set_type                              8             bslbf
            language_set_name_length                        8             uimsbf
            language_set_name       8*language_set_name_length            bslbf
        }
}
```

【図12】

```
Syntax                                                  Number of Bits    Mnemonic
text_item () {
        text_item_length                                    16            uimsbf
        text_item_id                                        16            uimsbf
        text_item_sub_id                                    16            uimsbf
        flags                                               8             bslbf
        number_of_used_language_sets                        8             uimsbf
        // loop for each language set
        for (i=0; i<number_of_used_language_sets; i++) {
            language_set_id                                 8             uimsbf
            reserved                                        4             bslbf
            text_string_length                              16            uimsbf
            text_string                   8*text_string_length            bslbf
            bitman ()
        }
        stuffing_bytes                                      8*n           bslbf
}
```

【図29】

```
/------MPEGAV
        |       --STREAMS_003
        |       |       --CHUNK 0031. MPEG2
```

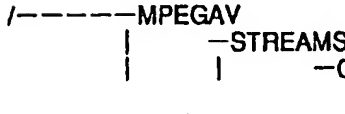
【図14】

Syntax	Number of Bits	Mnemonic
album () {		
album_length	32	uimsbf
reserved	6	bslbf
volume_status	1	bslbf
if (volume_status== "1b") {		
chief_volume_flag	1	bslbf
} else {		
reserved	1	"0"
}		
if (volume_status== "1b") {		
if (chief_volume_flag== "1b") {		
reserved	6	bslbf
album_type	2	bslbf
albim_id	128	bslbf
number_of_discs_in_album	16	uimsbf
number_of_volumes_in_album	16	uimsbf
for (i=0; i<number_of_volumes_in_album; i++) {		
disc_id_for_album_member	128	bslbf
volume_id_for_album_member	128	bslbf
title_offset_number	16	uimsbf
}		
reserved_for_program_bind	8	bslbf
number_of_program_binds	8	uimsbf
for (i=0; i<number_of_program_binds; i++) {		
number_of_program_in_this_program_bind	16	uimsbf
for (i=0; i<number_of_programs_in_this_program_binds; i++) {		
disc_id_for_program_bind_member	128	uimsbf
volume_id_for_program_bind_member	128	uimsbf
program_number	16	uimsbf
}		
} else { // chief_volume_flag== "0b"		
chief_disc_id	128	uimsbf
chief_volume_id	128	uimsbf
(album_id	128	bslbf
}		
}		
}		

【図17】

Syntax	Number of Bits	Mnemonic
PROGRAM_$$$. PGI {		
file_type_id	8*16	char[16]
program ()		
text_block ()		
}		

【図32】



【図16】

Syntax	Number of Bits	Mnemonic
title_info () {		
title_info_length	32	uimsbf
flags_for_title	32	bslbf
cgit_file_id	16	uimsbf
title_start_chunk_group_time_stamp	64	uimsbf
title_end_chunk_group_time_stamp	64	uimsbf
title_playback_time ()	32	bcd
reserved	32	bslbf
number_of_marks	16	uimsbf
for (i=0; i<number_of_marks; i++) {		
reserved	4	bslbf
mark_type	4	bslbf
relative_time_stamp_in_title	64	uimsbf
}		
stuffing_bytes	8*n	bslbf
}		

【図18】

Syntax	Number of Bits	Mnemonic
program () {		
program_length	32	uimsbf
flags_for_program	32	bslbf
program_status	4	bslbf
program_playback_time ()	32	bcd
reserved	32	bslbf
number_of_play_sequences	16	uimsbf
for (j=0; j<number_of_play_sequence; j++) {		
number_of_play_lists	16	uimsbf
for (k=0; k<number_of_play_lists; k++) {		
play_list_start_time_stamp_offset	64	uimsbf
play_list (k)		
}		
}		
stuffing_bytes	8*n	bslbf
}		

【図21】

Syntax	Number of Bits	Mnemonic
CHUNKGROUP_###. CGIT {		
file_type_id	8*16	char[16]
chunkgroup_time_base_flags	32	bslbf
chunkgroup_time_base_offset	64	uimsbf
chunk_connection_info ()		
text_block ()		
}		

【図19】

Syntax	Number of Bits	Mnemonic
play_list () {		
// playback sequence of play items in this play list		
number_of_play_items	16	uimsbf
for (k=0; k<number_of_play_items; k++) {		
play_item_number	16	uimsbf
reserved	31	bslbf
seamless_connection_flag	1	bslbf
}		
// play_item_table		
for (PIN=1; PIN<=number_of_play_items_in_program; PIN++) {		
play_item ()		
}		
}		

【図20】

Syntax	Number of Bits	Mnemonic
play_item () {		
play_item_length	32	uimsbf
play_item_type	8	bslbf
play_mode	8	bslbf
total_playback_time ()	32	bcd
menu_item_number	16	uimsbf
return_item_number	16	uimsbf
next_item_number	16	uimsbf
prev_item_number	16	uimsbf
if (play_item_type= "0000b") {		
// play item for one "cut"		
title_number	16	uimsbf
// IN point		
item_start_time_stamp	64	uimsbf
// OUT point		
item_end_time_stamp	64	uimsbf
}		
}		

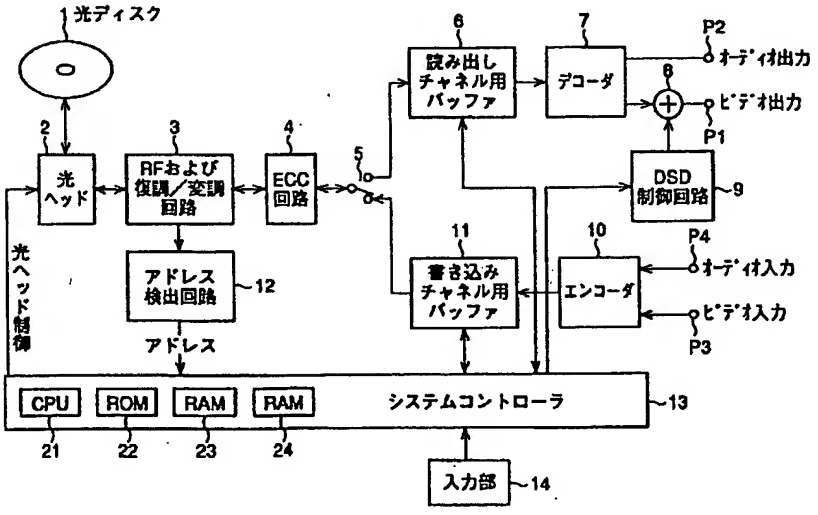
【図22】

Syntax	Number of Bits	Mnemonic
chunk_connection_info () {		
chunk_connection_info_length	32	uimsbf
reserved	16	bslbf
number_of_chunks	16	uimsbf
chunk_sync_play_flag	8	bslbf
// chunk info file list		
for (i=0; i<number_of_chunks; i++) {		
chunk_arrangement_info ()		
}		
}		

【図２３】

Syntax	Number of Bits	Mnemonic
chunk_arrangement_info () {		
chunk_arrangement_info_length	32	uimsbf
chunk_info_file_id	16	bslbf
reserved	5	bslbf
chunk_switch_stream_id	16	bslbf
presentation_start_cg_time_count	64	uimsbf
presentation_end_cg_time_count	64	uimsbf
reserved	4	bslbf
chunk_time_count_type	4	bslbf
number_of_start_original_time_count_extension	8	uimsbf
number_of_end_original_time_count_extension	8	uimsbf
// presentation start position and time		
presentation_start_original_time_count	64	uimsbf
presentation_end_original_time_count	64	uimsbf
for (j=0; j<number_of_start_original_time_count_extension; j++)		
tc_ext_attributes	16	bslbf
start_original_time_count_extension	64	uimsbf
}		
// presentation end position and time		
for (k=0; k<number_of_end_original_time_count_extension; k++) {		
tc_ext_attributes	16	bslbf
end_original_time_count_extension	64	uimsbf
}		
transition_info ()		
}		

【図２５】



【図３６】

field name	value
file_type_id	"TITLE_INFO_00000"

【図３８】

field name	value
file_type_id	"CHUNKGROUP_0000"

【図24】

```
Syntax                                              Number of Bits    Mnemonic
CHUNK_%%%%. ABST {
    file_type_id                                    8*16              char[16]
    reserved                                        4                 bslbf
    chunk_file_id                                   16                uimsbf
    info_type                                       4                 bslbf
    // stream_info ()

    if (info_type== "MPEG2_System_TS" ) {
        number_of_programs                          8                 uimsbf
    else {
        number_of_programs                          8                 "0000 0001"
    }
    for (i=0; i<number<of_programs; i++) {
        number_of_streams                           8                 uimsbf
        for (i=0; i<number_of_streams; i++) {
            stream_identifier                       8                 bslbf

            // slot type information
            reserved                                4                 bslbf
            slot_unit_type                          4                 bslbf
            if (slot_unit_type== "time_stamp" ) {
                slot_time_length                    32                uimsbf
            } else {
                reserved                            32                bslbf
            }
            number_of_slots                         32                uimsbf
            reserved                                4                 bslbf
            switch (info_type) {
                case MPEG1_System :
                case MPEG2_System_PS :
                case MPEG2_System_TS :
                case video_elementary_stream
                        number_of_I_pictures_in_a_slot  4             uimsbf
                        break ;
                default :
                        reserved                    4                 bslbf
                        break ;
            }

            // stream attribute
            ES_attribute ()
        }

        // loop of slot info
        for (i=0; i<number_of_streams; i++) {
            for (i=0; i<number_of_slots; i++) {
                slot_info ()
            }
        }
    }
}
```

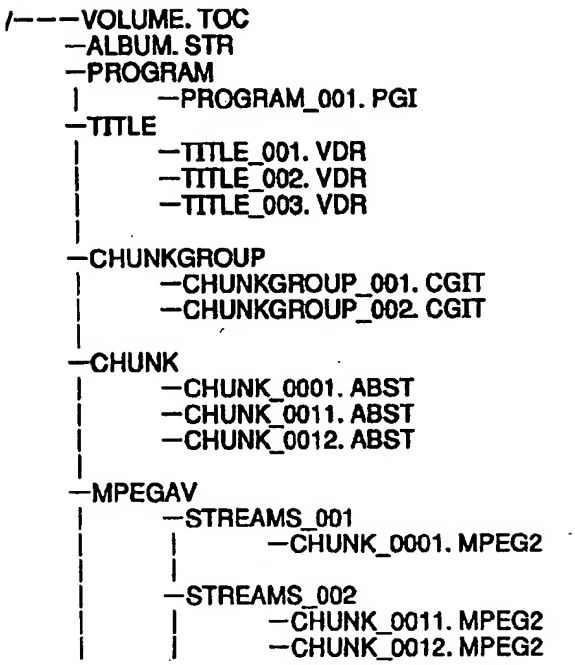
【図39】

chunk_sync_play_flag	Meaning
0b	play only one chunk at the same time
1b	need to play chunks simultaneously

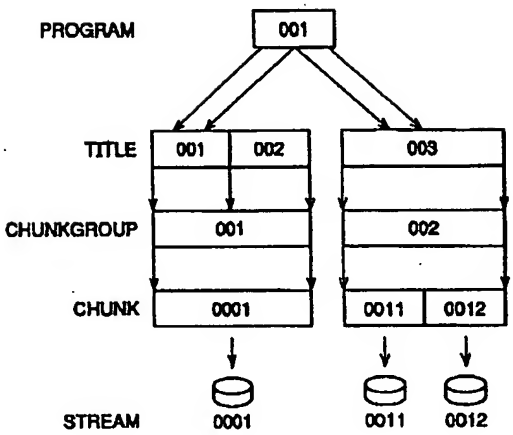
【図41】

field name	value
file_type_id	"STREAM_INFO_0000"

【図２６】



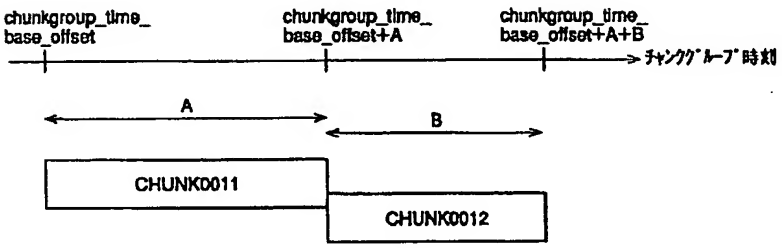
【図２７】



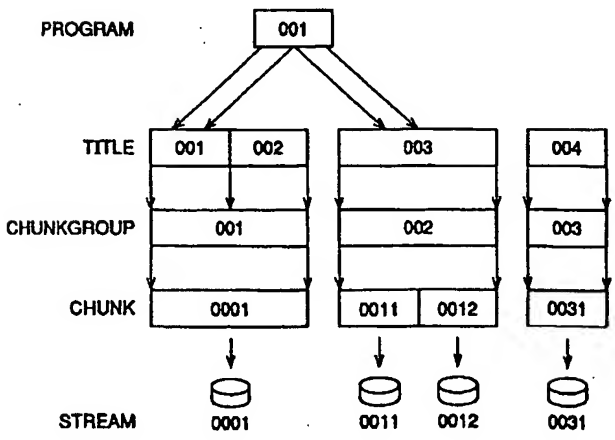
【図４４】

field name	value
file_type_id	"PROGRAM_INFO_000"

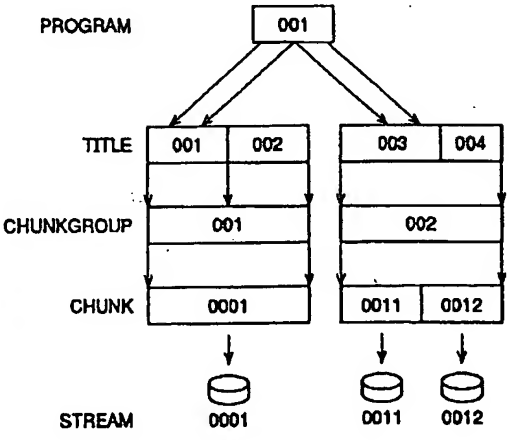
【図２８】



【図３１】



【図３５】



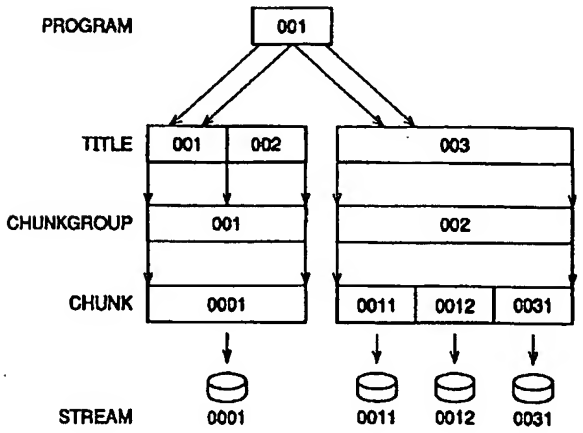
【図30】

```
/---VOLUME.TOC
    -ALBUM.STR
    -PROGRAM
    |     -PROGRAM_001.PGI
    -TITLE
    |     -TITLE_001.VDR
    |     -TITLE_002.VDR
    |     -TITLE_003.VDR
    |     -TITLE_004.VDR*
    |
    -CHUNKGROUP
    |     -CHUNKGROUP_001.CGIT
    |     -CHUNKGROUP_002.CGIT
    |     -CHUNKGROUP_003.CGIT*
    |
    -CHUNK
    |     -CHUNK_0001.ABST
    |     -CHUNK_0011.ABST
    |     -CHUNK_0012.ABST
    |     -CHUNK_0031.ABST*
    |
    -MPEGAV
    |     -STREAMS_001
    |     |     -CHUNK_0001.MPEG2
    |     |
    |     -STREAMS_002
    |     |     -CHUNK_0011.MPEG2
    |     |     -CHUNK_0012.MPEG2
    |     |
    |     -STREAMS_003*
    |     |     -CHUNK_0031.MPEG2*
    |     |
```

【図33】

```
/---VOLUME.TOC
    -ALBUM.STR
    -PROGRAM
    |     -PROGRAM_001.PGI
    -TITLE
    |     -TITLE_001.VDR
    |     -TITLE_002.VDR
    |     -TITLE_003.VDR
    |     -TITLE_004.VDR*
    |
    -CHUNKGROUP
    |     -CHUNKGROUP_001.CGIT
    |     -CHUNKGROUP_002.CGIT
    |
    -CHUNK
    |     -CHUNK_0001.ABST
    |     -CHUNK_0011.ABST
    |     -CHUNK_0012.ABST
    |     -CHUNK_0031.ABST*
    |
    -MPEGAV
    |     -STREAMS_001
    |     |     -CHUNK_0001.MPEG2
    |     |
    |     -STREAMS_002
    |     |     -CHUNK_0011.MPEG2
    |     |     -CHUNK_0012.MPEG2
    |     |     -CHUNK_0031.MPEG2*
    |     |
```

【図34】



【図37】

mark_type	Meaning
0000b	Index type 1
0001b	Index type 2
0010b .. 0111b	Index type 3 .. Index type 8
1000b	skip IN
1001b	skip OUT
1010b	jump to title end or next title
1011b	scene change
1100b	audio mute
1101b	audio peak
1110b	picture still
1111b	reserved

Index : direct entry point in the title
skip IN : start point of title skip
skip OUT : end point of title skip

【図43】

slot_unit_type	Meaning
0000b	'time_stamp' : time stamp value
0001b	'GOP' : one GOP(Group of pictures)
0010b	'audio_frame' : one audio frame
0011b .. 1111b	reserved

【図40】

original_time_count_type	Meaning
0000	MPEG2 System time stamp(90kHz)
0001	SMPTE timecode
0010	field(525/60)(59.94Hz)
0011	field(625/50)
0100	frame(525/60)(29.97Hz)
0101	frame(625/50)
0110	drop frame time code(525/60)
0111	non drop frame time code(525/60)
1000	60Hz
1001	50Hz
1010	24Hz
1011	second
1100 .. 1111	reserved

【図42】

Info_type	Meaning
0000	MPEG2_System_PS
0001	MPEG2_System_TS
0010	MPEG2_System_PES
0011	video_elementary_stream
0100	elementary_stream_except_video
0101	MPEG1_System_stream
0110	reserved
0111	reserved
1000	Consumer_DVC
1001 .. 1111	reserved

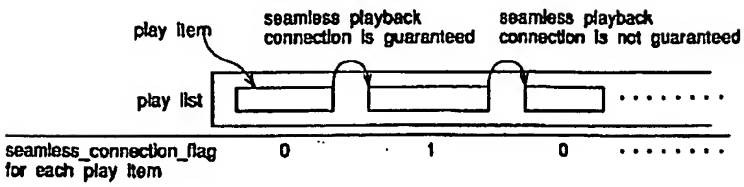
【図45】

program_status	Meaning
0000b	none
0001b	original
0010b	copy
0011b	preview
0100b	rehearsal
0101b	temporary
0110b	complete
0111b	broken
1000b	editing
1001b	backup
1010b .. 1111b	reserved

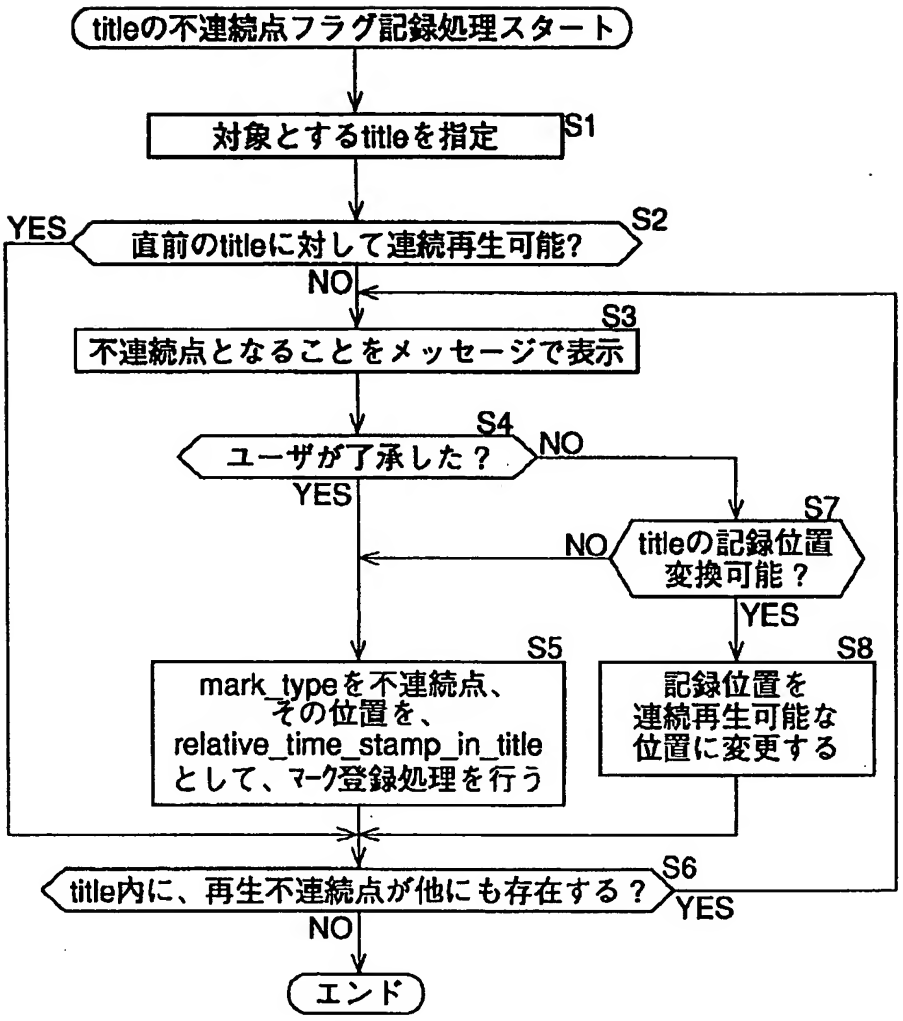
【図46】

seamless_connection_flag	value
0b	seamless playback connection with the previous play item is not guaranteed or unknown
1b	seamless playback connection with the previous play item is guaranteed

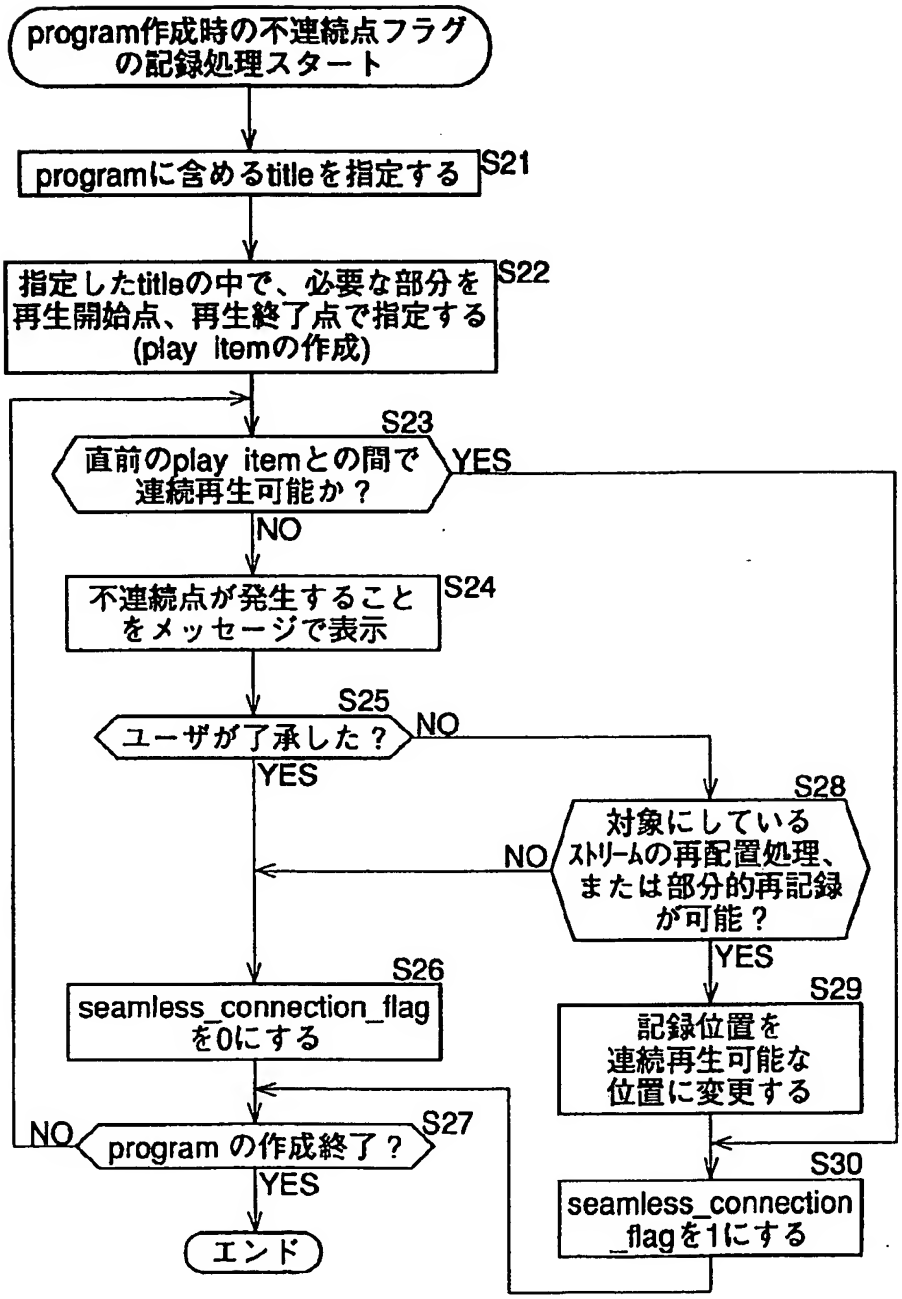
【図47】



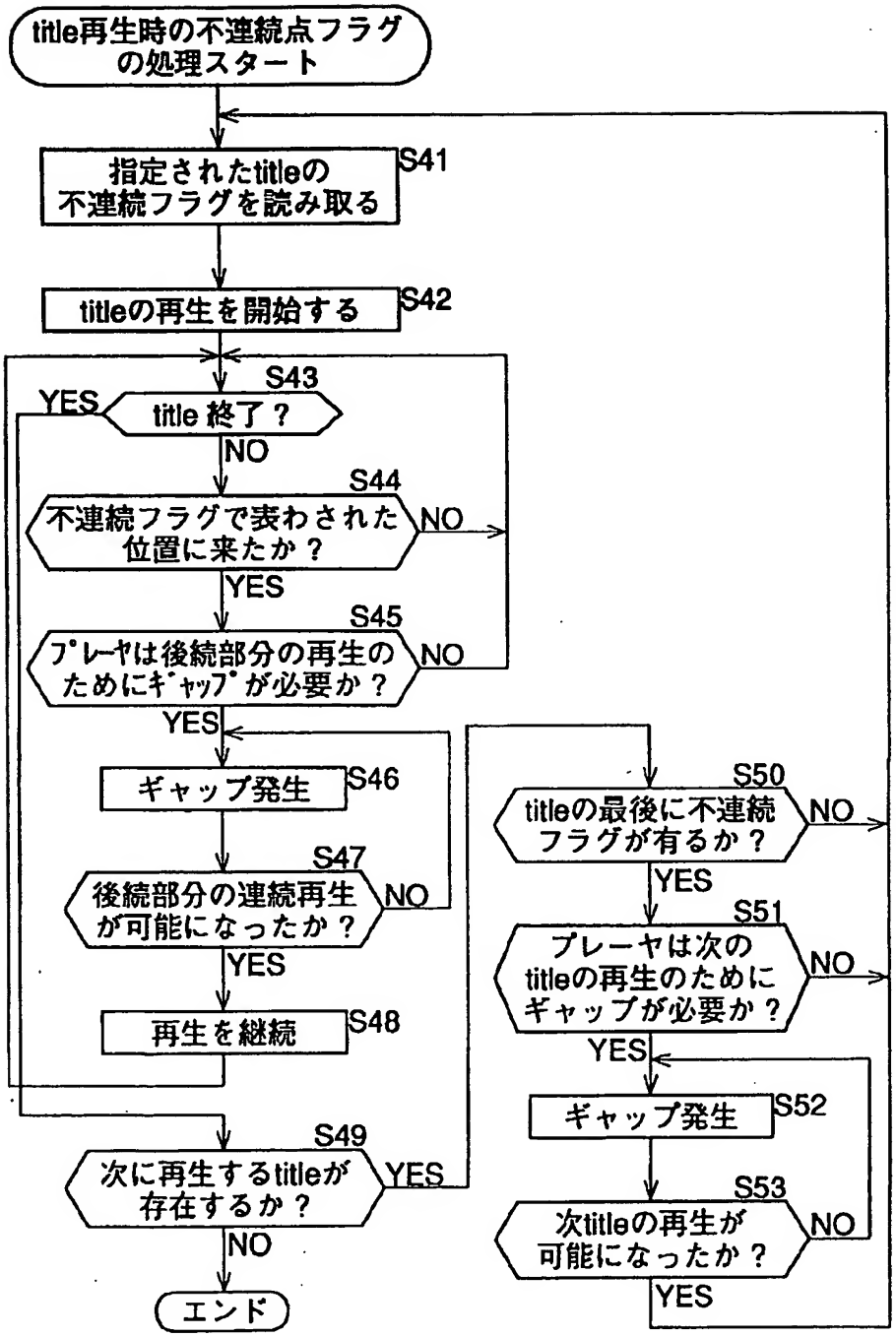
【図４８】


titleの不連続点フラグ記録処理スタート
対象とするtitleを指定
S1
YES
直前のtitleに対して連続再生可能?
S2
NO
S3
不連続点となることをメッセージで表示
S4
ユーザが了承した？
NO
YES
S7
NO
titleの記録位置
変換可能？
YES
S5
mark_typeを不連続点、
その位置を、
relative_time_stamp_in_title
として、ﾏｰｸ登録処理を行う
S8
記録位置を
連続再生可能な
位置に変更する
S6
title内に、再生不連続点が他にも存在する？
YES
NO
エンド

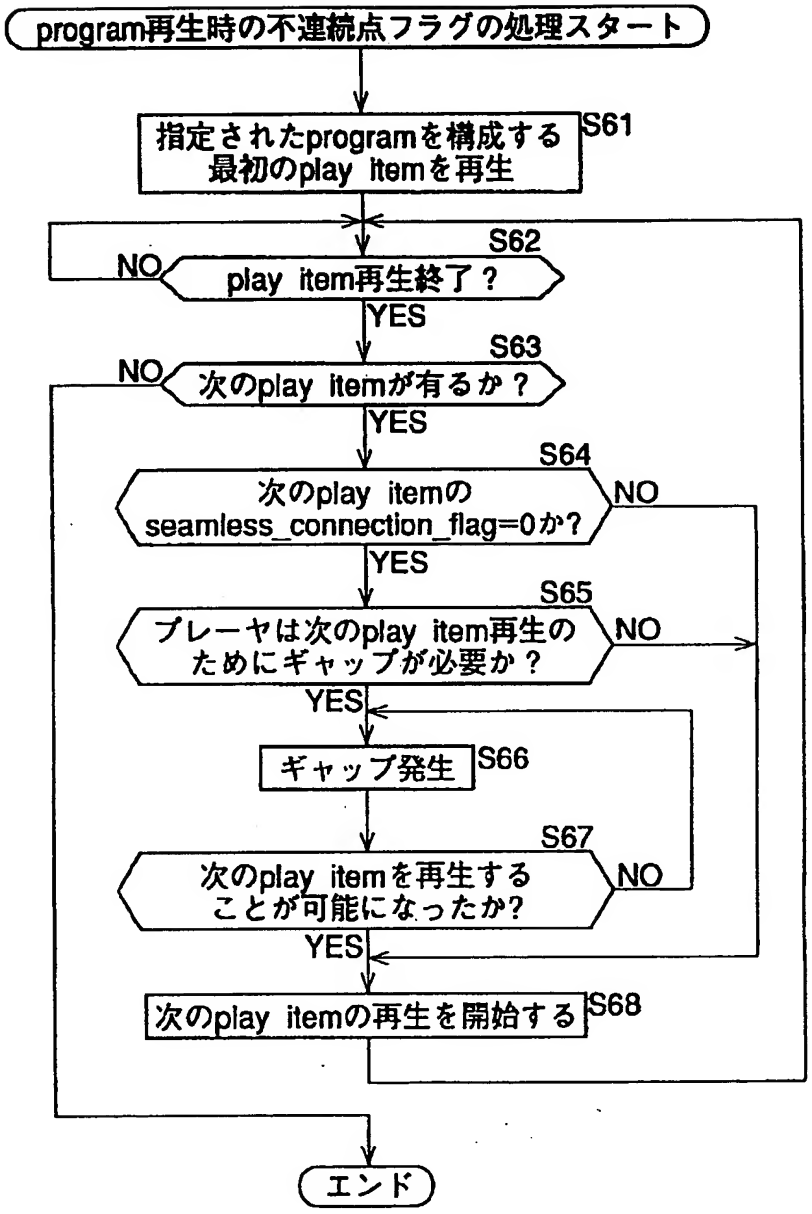
【図４９】


program作成時の不連続点フラグの記録処理スタート
programに含めるtitleを指定する
S21
指定したtitleの中で、必要な部分を再生開始点、再生終了点で指定する(play itemの作成)
S22
S23
直前のplay itemとの間で連続再生可能か？
YES
NO
不連続点が発生することをメッセージで表示
S24
S25
ユーザが了承した？
NO
YES
S28
対象にしているストリームの再配置処理、または部分的再記録が可能？
NO
YES
S26
seamless_connection_flagを0にする
S29
記録位置を連続再生可能な位置に変更する
NO
program の作成終了？
S27
YES
S30
seamless_connection_flagを1にする
エンド

【図５０】


title再生時の不連続点フラグの処理スタート
指定されたtitleの不連続フラグを読み取る
S41
titleの再生を開始する
S42
S43
YES
title 終了？
NO
S44
不連続フラグで表わされた位置に来たか？
NO
YES
S45
ﾌﾟﾚｰﾔは後続部分の再生のためにｷﾞｬｯﾌﾟが必要か？
NO
YES
ギャップ発生
S46
S47
後続部分の連続再生が可能になったか？
NO
YES
再生を継続
S48
S49
次に再生するtitleが存在するか？
YES
NO
エンド
S50
titleの最後に不連続フラグが有るか？
NO
YES
S51
プレーヤは次のtitleの再生のためにギャップが必要か？
NO
YES
ギャップ発生
S52
S53
次titleの再生が可能になったか？
NO
YES

【図５１】


program再生時の不連続点フラグの処理スタート
指定されたprogramを構成する最初のplay itemを再生
S61
S62
NO
play item再生終了？
YES
S63
NO
次のplay itemが有るか？
YES
S64
次のplay itemのseamless_connection_flag=0か?
NO
YES
S65
プレーヤは次のplay item再生のためにギャップが必要か？
NO
YES
ギャップ発生
S66
S67
次のplay itemを再生することが可能になったか?
NO
YES
次のplay itemの再生を開始する
S68
エンド